POLITECNICO DI TORINO

Master of Science in Mechatronic Engineering

Master Thesis

Automated generation of stable bias-dependent small-signal behavioural macromodels for circuit-level simulation

Supervisor
Stefano Grivet-Talocia

Candidate
Marco DE STEFANO
ID number 233452

Academic Year 2017-2018

To my family and to those who supported me during this journey

Acknowledgements

I would first like to thank my thesis supervisor Stefano Grivet-Talocia for the passion that he transmits during his work and for using the same care with the students. His attitude is an unmeasurable source of inspiration, and I am grateful for the opportunity of learning under his guidance. I hope to have the chance of continuing knocking at his door, which was always patiently open.
A grateful appreciation to the two colleagues, Alessandro and Tommaso, which whom I had the possibility of sharing this incredible journey. I would like to thank them for their patience and passion: they have inspired me to achieve the best. Without their effort this work would not be the same. I wish them to reach all their goals: I am sure that with your skills you could do whatever you want to.
Thank you!

Marco De Stefano

Summary

The system-level verification of signal and power integrity of modern electronic systems such as mobile phones or tablets is still a challenge even for most advanced manufacturers, mainly due to the required time-consuming and memory-demanding numerical simulations. This fact led in recent years to the development of various model simplification approaches sometimes denoted as model order reduction, behavioural or surrogate modelling, and macromodeling. The main idea is to represent complex systems and structures with numerical simulation models that capture the main features of their electrical responses that are relevant to the application at hand, and neglect unimportant effects.

Macromodeling is a mature field as applied to electrical interconnects and, in general, to linear and time-invariant systems. Even if these structures are responsible for all electromagnetic coupling and interference, the latter effects must be quantified by numerical simulations that account also for the digital, analog, and mixed-signal circuit blocks that send and receive signals, and that are connected to the global power distribution network. A description of such devices based on a fully-detailed transistor-level makes system-level simulation unfeasibly due to overwhelming complexity.

This work proposes a surrogate modelling approach for those passive or active circuit blocks that are specifically designed to operate as linearly as possible around a given operating point. Examples are Low Noise Amplifiers (LNA), Operational Amplifiers (OA), Low Dropout regulators (LDO), programmable filters, and even individual components such as integrated inductors and capacitors whose inductance and capacitance can be tuned by changing bias conditions. Our approach approximates the dynamic behaviour of these components as linear state-space systems, whose state-space matrices depend on external parameters, such as the nominal bias voltage. This approach allows an accurate representation of signal and power noise, including the sometimes difficult to represent signal-to-power noise coupling paths.

The models are constructed starting from small-signal scattering responses, obtained by numerical simulation of the original system at different bias points. A parameterized Generalized Sanathanan-Koerner iteration is used to identify the model coefficients. One of the main results of this thesis is the inclusion in the identification algorithm of a set of constraints that are able to enforce by construction the uniform stability of the bias-dependent model poles, for all values of the bias within its design range. A second main result is a fully automated synthesis of a SPICE equivalent network that realizes the parameterized model in a form that can be simulated by off-the-shelf circuit solvers. This synthesis is based on a parameterization of the bias and in general parameter dependence in terms of multivariate orthogonal polynomials.

Supported by an industrial partner (Intel), we tested our algorithm on several test cases. The proposed approach was always able to construct very efficient and accurate models for all the structures that were analyzed, which ranged from simple integrated inductors and capacitors to more complex LDO's and Operational Amplifier circuit blocks. Our future activities will be devoted to the integration of this method into well-established industrial design flows, always in collaboration with industrial partners, in order to demonstrate the feasibility of an all-macromodel-based fast power integrity verification framework.

Contents

Preface

The present thesis project is self-consistent and it has been developed independently by the Author, under the supervisor guidance. However, the results that have been achieved are of practical interest only if cast in a more general framework, in which other two thesis projects are involved: the shared effort of a team composed of Tommaso Bradde, Marco De Stefano and Alessandro Zanco enabled each member of the group to finalize his work. Being part of a joint effort, each of the three thesis projects shares a common background, which has been summarized in Chapters 1 and 2. These two chapters were written jointly and are common to all three thesis projects. The remaining chapters of each dissertation are the core of each project and are original for each individual team member.

Chapter 1

General Framework and Motivations

This Chapter is co-authored by T.Bradde, M. De Stefano and A. Zanco.

1.1 Data Driven Modeling

This thesis project concerns mathematical modeling of linear dynamic systems, namely, systems that are governed by linear differential equations. By "mathematical modeling", we refer to the procedure by means of which a representation of a physical phenomenon or structure is given in a numerically (i.e quantitatively) exploitable form. This kind of representation grants us the opportunity to describe and predict what would happen in a given scenario in which the described object is involved; we can say that such a procedure is at the same time the foundation and the objective of science and a necessary step of the design process in every engineering field.
Although the first-principle laws of science are theoretically able to properly describe a broad range of dynamic phenomena, usually making use of partial differential equations, it is often inappropriate or impossible to derive from them a model able to satisfy the requirements of a current design process: the (exponentially) increasing complexity of the structures to be modeled would lead to an excessive computational cost with respect to the need of an easily manageable description of the item under design. Further, a model derived from first-principle laws must take care of all the physical quantities involved in the system dynamic, while often, only a subset of them is of practical interest.
The Data Driven Modeling techniques are intended to overcome these issues and to provide simpler yet accurate descriptions, able to catch the case-relevant aspects of the structures under investigation by exploiting, as common ground, a set of data collected to extract information about the system behaviour. Making use of proper algorithms, a suitable reconstruction of the relations underlying the data is achieved.
To gather the data, one can either carry out physical measurements of the desired quantities to be tracked or perform (once) a set of first-principle simulations from which the simplified model can be derived.
The most appropriate algorithm to process the data is always a matter of purposes, since

the structure of the algorithm influences, in some measure, the structure of the final model.
Beside the possible implementations, a broad spectrum classification of those algorithms can be based upon the a priori assumptions about the structure of the system: in the so called white and gray box approaches, a total or partial knowledge of the structure is assumed and the algorithm is expected to give back some quantities that characterize the imposed structure from the physical point of view; on the other hand, black box approaches make no assumptions on this structure and make no claims towards a physical description of the system, focusing only into the construction of models that fit numerically the data of the input-output relationship.
The first class of methods can give a deeper insight into the system behaviour, but they rely on the goodness of the a priori assumptions, that can result to be inaccurate or not possible to be made at all. Conversely, the lack of physical meaning of a black box model is counterbalanced by the opportunity to derive an input-output description without any assumption beyond linearity.
From now on, we will treat the black box methods and we will refer to the obtained model as "Macromodel".

1.1.1 Macromodels: Construction Flow and Advantages

In the following, we will focus on macromodels devoted to the behavioural simulation of complex electrical interconnects, or, more generally, electromagnetic structures. The main objective of the macromodeling procedure is to obtain a macromodel that replaces the high complexity dynamics of the structure with a lower complexity model, which catches only the main features of the relationship between the electrical inputs and outputs of interest.
If we are modeling the system in the frequency domain, our starting point is a set of input-output data:

$$\breve{H}_k = \breve{H}(s_k) \quad for \quad k = 1,2,...,K \tag{1.1}$$

where s_k denotes a complex frequency and $\breve{H}(s_k)$ is the transfer function of the system sampled at s_k. The total number of measurements is K.
In most cases, the measurements are performed at the real frequencies $\mathrm{j}\omega_k$, with $s_k = \mathrm{j}\omega_k$. In this case we have:

$$\mathrm{j}\omega_1 = \mathrm{j}\omega_{min}, \quad \mathrm{j}\omega_K = \mathrm{j}\omega_{max} \tag{1.2}$$

The objective is then to reconstruct the response by means of an interpolation or approximation procedure that returns a model:

$$H(\mathrm{j}\omega) \approx \breve{H}(\mathrm{j}\omega) \quad for \quad \omega \in [\omega_{min}, \omega_{max}] \tag{1.3}$$

Throughout this text, we will denote with the symbol $\breve{H}(\cdot)$ the true system response, while with the symbol $H(\cdot)$ the model response. The obtained model is intended to be exploited in a circuit simulation software such as SPICE or EMTP in a fast and reliable way.
We now present a brief overview of how a macromodel is usually obtained and of the strong points that makes it useful.

- **Macromodeling from field solver data**: a full-wave solver is used to obtain the input-output data; detailed knowledge of the structures and of the characteristics of the actual system is required to perform the primary simulation. The data can be collected both in the time domain or in the frequency domain.
  This method is not properly a black-box one, since the structure of the model must be known to perform the full-wave simulation; anyway, we can say that it is a black box method for what concerns the macromodeling algorithm, that receives only data as inputs, without additional informations about structure. This scenario is common in industrial design environments where commercial field solvers are used.

- **Macromodeling from measurements**: a physical realization of the system under modeling is provided; the data are collected and reconstructed by performing measurements over the electrical ports that we wish to characterize. Also in this case, both frequency and time domain data can be gathered. This approach is truly black-box, in every step of the identification procedure.

Once the data are processed by the chosen algorithm, one can dispose of the obtained macromodel with the following advantages:

1. A closed form expression for the behaviour of the system is obtained from the discrete set of data points collected.

2. The macromodel describes the system behaviour without disclosing any insight about the physical structure: sharing a macromodel doesn't represent a risk for the diffusion of proprietary information.

3. Whatever is the nature of the data set used for the fitting, the resulting macromodel is intended to permit fast time domain simulations.

4. The obtained macromodel can be interfaced with other macromodels for simulation of large interconnects system, allowing the possibility to simulate and optimize various design scenarios.

1.1.2 Macromodel Requirements for Simulations

Some features are required on the macromodel, in order to guarantee its exploitability and reliability. In particular, since we are dealing with the modeling of linear systems, a suitable model structure should be chosen among all the possible ones; indeed, we know that when a system is governed by ordinary differential equations, all the transfer functions that can be derived for its input-output description result to be rational functions of the Laplace variable s.

The choice of a model structure of this type not only catches the underlying governing laws of the system, but results also particularly appropriate to be exploited to perform simulations driven by linear circuit simulation software.
The numerical precision of the model must always be consistent with some physical characteristics of the modeled structure to reproduce its behavior correctly; here, we list the most relevant in an intuitive fashion, leaving a more precise description to later sections.

- **Realness.** Although the rational macromodels make use of complex variables to describe the input-output behaviours, all the simulated quantities must be real numbers when observed in the time domain.

- **Causality.** Any physical system at rest can change its state only as a result of an external stimulus; for an input-output description, this fact implies the necessity of the output to be temporally preceded by its cause, the input.

- **Stability.** The concept of stability can be provided with various definitions; in the following we define stable a model whose poles show negative real part, that is, if $\{p_i\}$ is the set of poles of the model, then:

$$\mathrm{Re}\,\{p_i\} < 0 \quad for \quad i = 1,2,...,n \tag{1.4}$$

  where n is the order of the associated transfer function. The lack of the stability property can imply numerically unbounded simulations that clearly do not reflect the behavior of a real system.

- **Passivity.** A system is passive if it is not able to generate energy on its own; it can realease energy to the outer environment only if that energy was previously provided and stored inside it. The property of passivity can be regarded as the most general, since it automatically implies stability, causality and realness.

1.1.3 Rational Fitting Algorithms

The choice of a particular fitting strategy is the first step in any modeling procedure: we must first fix the structure of our model in order to restrict the set of all the possible candidate representations. Since our aim is to model electrical interconnects and their frequency-dependent behavior, the system will intrinsically exhibit a linear relationship between input and output, due to the nature of the electromagnetic phenomena.
It is well known that any linear system is fully characterized by a rational function of the complex variable s through its input-output transfer function:

$$H(s) = \frac{N(s)}{D(s)}, \tag{1.5}$$

Where $N(s)$ and $D(s)$ are polynomials. Therefore, a natural choice is to try to reconstruct the system through a rational fitting procedure, that returns a model potentially able to catch all the information of interest.

Rational fitting algorithms make use of rational functions as basis for the model. Rational functions are universal approximators: any set of data can be fitted by a series of rational functions if a suitable order (i.e. number of basis functions) is considered. Even if this is for sure an encouraging starting point, several issues affect a modeling process relying on rational fitting:

- The behavior of the returned model is very accurate at the fitting points, but might show an unwanted and improper oscillating nature between the data points and beyond the limits of the data interval; this is particularly common when a very high

order for the interpolating function is chosen.
This phenomenon is known as over-fitting and must be taken into account during the identification procedure: one should use a subset of the available data to test the model quality at points of the domain that are not exploited for the fitting procedure.

- The imposition of constraints that ensure the model physical consistency can prevent the convergence of rational fitting algorithms or, most often, be the cause of a poor quality of the fitting.

From now on, we will assume that the model to be identified is a proper rational function of the variable s, although an extension to the improper case is straightforward. The unknowns that the rational fitting algorithm is intended to return depend on the formulation of the rational function that we want to use. This formulation is fundamental because, as we will see, it can cast the model in forms that are more suitable with respect to others to achieve a good approximation. We now present the most common formulations of rational functions together with the unknowns that an algorithm is expected to return when such formulations are used as starting point.

- **Ratio of polynomials**: in this case, we assume that the model is representative of an underlying dynamics expressed as:

$$H(s;\boldsymbol{x}) = \frac{N(s;\boldsymbol{x})}{D(s;\boldsymbol{x})} = \frac{a_0 + a_1 s + a_2 s^2 + \cdots + a_m s^m}{b_0 + b_1 s + b_2 s^2 + \cdots + b_{n-1} s^{n-1} + s^n} \tag{1.6}$$

in this case, the unknown vector $\boldsymbol{x}$ collects the $2n$ parameters:

$$\boldsymbol{x} = (a_0, a_1, a_2, \ldots, a_m, b_0, b_1, b_2, \ldots, b_{n-1})^T \tag{1.7}$$

and the quality of the fitting can be evaluated by means of the residual quantity

$$r_k(\boldsymbol{x}) = \breve{H}_k - \frac{a_0 + a_1 s_k + a_2 s_k^2 + \cdots + a_m s_k^m}{b_0 + b_1 s_k + b_2 s_k^2 + \cdots + b_{n-1} s_k^{n-1} + s_k^n} \tag{1.8}$$

evaluated for each of the data samples.

- **Pole-zero form**: with this formulation the rational function reads

$$\breve{H}(s,\boldsymbol{x}) = \alpha \frac{\prod_{j=1}^{n-1}(s - z_j)}{\prod_{j=1}^{n}(s - p_j)}; \tag{1.9}$$

the $2n$ unknown vector is now:

$$\boldsymbol{x} = (\alpha, z_1, z_2, \ldots, z_{n-1}, p_1, p_2, \ldots, p_n)^T \tag{1.10}$$

and each residual quantity is evaluated as:

$$r_k(\boldsymbol{x}) = \breve{H}_k - \alpha \frac{\prod_{j=1}^{n-1}(s - z_j)}{\prod_{j=1}^{n}(s - p_j)}. \tag{1.11}$$

- **Partial fractions form**: in this case, the rational function is expressed as a series of partial functions of the form:

$$H(s, \boldsymbol{x}) = \sum_{j=1}^{n} \frac{c_j}{s - p_j}, \tag{1.12}$$

with the assumption that the multiplicity of each pole equals one, that is:

$$p_i \neq p_j \quad \forall i \neq j. \tag{1.13}$$

The $2n$ unknown vector is now defined as:

$$\boldsymbol{x} = (c_1, c_2, \ldots, c_n, p_1, p_2, \ldots, p_n)^T, \tag{1.14}$$

and the residual quantities are:

$$r_k(\boldsymbol{x}) = \breve{H}_k - \sum_{j=1}^{n} \frac{c_j}{s - p_j}. \tag{1.15}$$

- **Ratio of rational functions**: to formulate the model in this form, we observe first that any rational function of the variable s can be expressed as a ratio of other two rational functions in s; for this reason, we can cast the model in a more general form that reads:

$$H(s; \boldsymbol{x}) = \frac{N(s; \boldsymbol{x})}{D(s; \boldsymbol{x})} = \frac{\sum_{i=1}^{n} c_i \varphi_i(s)}{\sum_{i=1}^{n} d_i \varphi_i(s)} \tag{1.16}$$

where both numerator and denominator are expressed as a sum of *rational basis functions* $\varphi_i(s)$. In this case, the unknowns vector embeds the $2n$ coefficients of the series expansions:

$$\boldsymbol{x} = (c_1, c_2, \ldots, c_n, d_1, d_2, \ldots, d_n)^T, \tag{1.17}$$

while the residual vector is defined as:

$$r_k(\boldsymbol{x}) = \breve{H}_k - \frac{\sum_{i=1}^{n} c_i \varphi_i(s)}{\sum_{i=1}^{n} d_i \varphi_i(s)} \tag{1.18}$$

1.2 Rational Fitting with Fixed Poles

Our main attempt is to formulate the rational fitting problem in such a way that a linear dependence holds between the unknowns and the basis functions that we want to use to fit the data. If this linear relation holds, then the rational fitting problem can be solved by means of a standard least squares problem: the basis functions are sampled in the points of the domain for which data points are available and the resulting numerical values are used to build the *regressor* matrix of the least square problem.
We can see how, among all the formulations of a rational function, the only one that can guarantee linearity between the unknowns and the basis functions is the partial fractions expansion (1.12) under the assumption that the poles p_j are fixed apriori. This formulation will be deeply exploited in the following since it allows the formulation of the rational fitting problem as a standard least squares problem.

1.2.1 Partial Fractions

We usually define the frequency-dependent basis functions, due to their very convenient numerical properties, as a set of partial fractions with a fixed set of poles. In particular, we realize a prescribed set of distinct $\bar{n}_r$ real poles $q_i \in \mathbb{R}^-$ and $\bar{n}_c$ complex pole pairs $q_{i,i+1} = q_i' \pm \mathrm{j}q_i'' \in \mathbb{C}^-$, where $\varphi_0(s) = 1$. The total number of basis functions is assumed to be $n = 1 + \bar{n}_r + 2\bar{n}_c$, including the constant term. We can define,

$$\begin{aligned} &\text{if } \bar{q}_i \in \mathbb{R} \rightarrow \varphi_i(s) = (\bar{s} - \bar{q}_i)^{-1}; \\ &\text{if } \bar{q}_i \in \mathbb{C} \rightarrow \begin{cases} \varphi_i(s) = (\bar{s} - \bar{q}_i)^{-1} \\ \varphi_{i+1}(s) = \varphi_i^*(s) = (\bar{s} - \bar{q}_i^*)^{-1} \end{cases} \end{aligned} \tag{1.19}$$

To improve numerical conditioning, this basis definition is based on normalized independent variables and poles throughout

$$\bar{s} = \frac{s}{\omega_0}, \quad \bar{q}_i = \frac{q_i}{\omega_0}, \tag{1.20}$$

where ω_0 is a scaling frequency, which is in general obtained considering the largest model pole.

1.2.2 Least Squares Formulation of the Fitting Problem

Denoting with $\varphi_i(s)$ the generic element of our basis of partial fraction defined over a set of poles $\{q_i\}$, with $i = 0,1,2,...,n$, then the residual quantities related to each data sample can be written as:

$$r_k(\boldsymbol{x}) = \breve{H}_k - \boldsymbol{\varphi}_k^T \boldsymbol{x} \tag{1.21}$$

with

$$\boldsymbol{\varphi}_k^T = (\varphi_1(s_k), \varphi_2(s_k), ..., \varphi_n(s_k)), \boldsymbol{x} = (c_1, c_2, ..., c_n)^T \tag{1.22}$$

If we drop the dependency of the residuals on $\boldsymbol{x}$ we can write the above relationship in matrix form by writing:

$$\begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_K \end{pmatrix} = \begin{pmatrix} \breve{H}_1 \\ \breve{H}_2 \\ \vdots \\ \breve{H}_K \end{pmatrix} - \begin{pmatrix} \boldsymbol{\varphi}_1^T \\ \boldsymbol{\varphi}_2^T \\ \vdots \\ \boldsymbol{\varphi}_K^T \end{pmatrix} \boldsymbol{x}. \tag{1.23}$$

We can use the more compact and general notation:

$$\boldsymbol{b} = \begin{pmatrix} \breve{H}_1 \\ \breve{H}_2 \\ \vdots \\ \breve{H}_K \end{pmatrix}, \quad \boldsymbol{r} = \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_K \end{pmatrix}, \quad \boldsymbol{\Phi} = \begin{pmatrix} \boldsymbol{\varphi}_1^T \\ \boldsymbol{\varphi}_2^T \\ \vdots \\ \boldsymbol{\varphi}_K^T \end{pmatrix} \tag{1.24}$$

and write:

$$\boldsymbol{r} = \boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x} \tag{1.25}$$

Since our goal is to minimize the value of the residuals, we can solve the least squares problem [26, 27]:

$$\boldsymbol{\Phi x} \approx \boldsymbol{b} \tag{1.26}$$

that returns an unknown vector $\boldsymbol{x}^*$ such that the euclidean norm of the vector $\boldsymbol{r}$ is minimized.
By writing the matrix $\boldsymbol{\Phi}$ in extended form we obtain the Cauchy matrix:

$$\boldsymbol{\Phi} = \begin{pmatrix} 1 & \frac{1}{s_1 - q_1} & \frac{1}{s_1 - q_2} & \dots & \frac{1}{s_1 - q_n} \\ 1 & \frac{1}{s_2 - q_1} & \frac{1}{s_2 - q_2} & \dots & \frac{1}{s_2 - q_n} \\ 1 & \vdots & \vdots & & \vdots \\ 1 & \frac{1}{s_K - q_1} & \frac{1}{s_K - q_2} & \dots & \frac{1}{s_K - q_n} \end{pmatrix}, \tag{1.27}$$

It is well known that the condition number [14, 22] of the normal equations associated to the regressor matrix,

$$\kappa(\boldsymbol{\Phi}) = \sqrt{\frac{\sigma_{max}(\boldsymbol{\Phi}^H \boldsymbol{\Phi})}{\sigma_{min}(\boldsymbol{\Phi}^H \boldsymbol{\Phi})}}, \tag{1.28}$$

strongly influences the quality of the solution of the least squares problem. Fortunately, being the partial fraction basis linearly independent (although not orthogonal) the Cauchy matrix is usually well conditioned.

1.3 General Rational Fitting

The situation explained in the previous section is desirable to solve the fitting problem, but it is very uncommon to known a priori the set of poles of the underlying system. For this reason, black box rational fitting algorithms must be able to return a model without any initial assumption beyond linearity. Two example of such algorithms are the Generalized Sanathanan-Koerner iteration (GSK), introduced in the following, and the Vector Fitting Iteration, for which a discussion can be found in [19].

1.3.1 Generalized Sanathanan Koerner Iteration

The GSK [32]iteration makes use of the model formulation (1.16) to iteratively solve a linearized version of the rational fitting problem. At each iteration ν of the algorithm a modified residual quantity, defined as:

$$r_k^{\nu}(\boldsymbol{x_\nu}) = \frac{D(s_k; \boldsymbol{x}_\nu)\breve{H}_k - N(s_k; \boldsymbol{x}_\nu)}{D(s_k; \boldsymbol{x}_{\nu-1})}; \quad for \quad k = 1, 2, .., K \tag{1.29}$$

is minimized in LS sense. In this formulation $D(s_k; \boldsymbol{x}_\nu)$ is the denominator of the model at the current iteration (that is the one that will be found after the solution of the LS problem), while $D(s_k; \boldsymbol{x}_{\nu-1})$ is the denominator of the model computed at the previous iteration, evaluated at the fitting points. We denote with $\boldsymbol{x}_\nu$ an iteration-dependent

unknowns vector.
The idea behind the GSK algorithm is that as the number of iteration increases, the estimate of the denominator stabilizes, implying that the residual quantity becomes for $\nu \to \infty$

$$r_k^{\nu}(\boldsymbol{x}_{\infty}) = \breve{H}_k - \frac{N(s_k;\boldsymbol{x}_{\infty})}{D(s_k;\boldsymbol{x}_{\infty})} \quad for \quad k = 1, 2, .., K, \tag{1.30}$$

which coincides with the residual that we actually want to minimize. When the model is cast in the form (1.16), then the components of the residual vector $\boldsymbol{r}^{\nu}(\boldsymbol{x}_{\nu})$ at iteration ν will read:

$$r_k^{\nu}(\boldsymbol{x}_{\nu}) = \frac{[\varphi_0(s_k) + \sum_{j=1}^{n} d_j^{\nu}\varphi_j(s_k)]\breve{H}_k - \sum_{j=0}^{n} c_j^{\nu}\varphi_j(s_k)}{\varphi_0(s_k) + \sum_{j=1}^{n} d_j^{\nu-1}\varphi_j(s_k)}. \tag{1.31}$$

Here we imposed $d_0 = 1$ to guarantee a unique solution of the system since the component φ_0 is usually associated with a constant term. We made all the coefficients iteration-dependent.
The iterative minimization of $||\boldsymbol{r}^{\nu}(\boldsymbol{x}_{\nu})||$ is achieved through the least square solution of the system:

$$(\mathbf{M}_{\nu-1}\boldsymbol{\Psi})\boldsymbol{x}_{\nu} \approx \mathbf{M}_{\nu-1}\boldsymbol{b} \tag{1.32}$$

where:

$$\mathbf{M}_{\nu-1} = diag\{m_1^{\nu-1}, m_2^{\nu-1}, ..., m_K^{\nu-1}\}, \quad m_k^{\nu-1} = \frac{1}{D(s_k;\boldsymbol{x}_{\nu-1})},$$

$$\boldsymbol{b} = (\breve{H}_1\varphi_0(s_1), \breve{H}_2\varphi_0(s_2), ..., \breve{H}_K\varphi_0(s_K))^T,$$

$$\boldsymbol{\Psi} = (\boldsymbol{\Phi}_0 - \breve{\mathbf{H}}\boldsymbol{\Phi}_1),$$

$$\boldsymbol{\Phi}_0 = \begin{pmatrix} \varphi_0(s_1) & \varphi_1(s_1) & ... & \varphi_n(s_1) \\ \varphi_0(s_2) & \varphi_1(s_2) & ... & \varphi_n(s_2) \\ \vdots & \vdots & & \vdots \\ \varphi_0(s_K) & \varphi_1(s_K) & ... & \varphi_n(s_K) \end{pmatrix} \tag{1.33}$$

$$\boldsymbol{\Phi}_1 = \begin{pmatrix} \varphi_1(s_1) & \varphi_2(s_1) & ... & \varphi_n(s_1) \\ \varphi_1(s_2) & \varphi_2(s_2) & ... & \varphi_n(s_2) \\ \vdots & \vdots & & \vdots \\ \varphi_1(s_K) & \varphi_2(s_K) & ... & \varphi_n(s_K) \end{pmatrix}$$

The rational basis functions used as a basis is often the partial fractions basis.
We end this chapter by pointing out that the formulations of GSK we presented is given for the scalar case; anyway, a straightforward extension is possible to the multiport systems. For details see [19]. From now on, we will denote with the symbol $\breve{\mathbf{H}}(\cdot) \in \mathbb{C}^{P\times P}$ the multiport response of the true system and with $\mathbf{H}(\cdot) \in \mathbb{C}^{P\times P}$ the multiport response of our models, where the symbol P denotes the number of ports of the system. In the following, we will describe the main model formulations used to characterize a multiport system macromodel.

1.4 Multiport (MIMO) Model Formulations

Approximating the true system response in a suitable macromodel form is fundamental to include the curve fitting result in system-level simulations using standard circuit solver such as SPICE. Several mathematical structures are available: the identification algorithm efficiency, in frequency and time domain, is affected by this choice.
In this Section we are going to describe the model formulation through a transfer matrix and a state space realization; the latter will be useful for the macromodel characterization.

1.4.1 Transfer Function Formulation

Recalling to the scalar model of (1.5), we extend now the formulation realizing a rational model of a MIMO system. Considering a generic MIMO LTI system with rational transfer function, we can adopt the so-called *Generalized Sanathanan-Koerner (GSK)* form [33] [19] as

$$\mathbf{H}(s) = \frac{\mathsf{N}(s)}{\mathsf{D}(s)} = \frac{\sum_{n=0}^{\bar{n}} \mathbf{R}_n \, \varphi_n(s)}{\sum_{n=0}^{\bar{n}} r_n \, \varphi_n(s)}, \tag{1.34}$$

where we denoted with $\mathbf{R}_{n,\ell} \in \mathbb{R}^{P \times P}$ and $r_{n,\ell} \in \mathbb{R}$ the numerator and denominator model (real-valued) coefficients, respectively.
Frequency variations are induced by chosen basis function $\varphi_n(s)$, which are rational functions of s , with $\bar{n}$ frequency basis order.
Avoiding an explicit parameterization of model poles is a critical and necessary condition: in fact, non-smooth behaviours may occur, e.g. when bifurcations are present, with a pair of coincident real poles that split into two complex conjugate poles, or viceversa (see [18] for details).
Both numerator and denominator of (1.34) share the same basis poles set, which are assumed stable.

1.4.2 State Space and Descriptor Form

We now explore the state space and descriptor realizations of a MIMO LTI system, starting from the pole-residue or GSK form of the model $\mathbf{H}(s)$ in the Laplace domain.

State Space for Pole-Residue Form

Considering a general $P \times P$ model in a pole-residue form, we can write

$$\mathbf{H}(s) = \mathbf{H}_\infty + \sum_{n=1}^{\bar{n}} \frac{\mathbf{R}_n}{s - q_n}, \tag{1.35}$$

where $\mathbf{H}_\infty = \mathbf{R}_0$ and $\bar{n}$ is the overall number of poles. We can denote

$$\mathbf{A} = \text{blkdiag}\{q_n \mathbb{I}_P\}_{n=1}^{\bar{n}_r} \tag{1.36}$$

$$\mathbf{B} = \big[1, \ldots, 1\big]^\mathsf{T} \otimes \mathbb{I}_P \tag{1.37}$$

$$\mathbf{C} = \begin{bmatrix} \mathbf{R}_1 & \cdots & \mathbf{R}_{\bar{n}} \end{bmatrix} \tag{1.38}$$

$$\mathbf{D} = \mathbf{H}_\infty \,. \tag{1.39}$$

Starting from the definitions of the matrices above, we can define a regular state-space realization of the system as

$$\begin{cases} \dot{\boldsymbol{x}}(t) = \mathbf{A}\boldsymbol{x}(t) + \mathbf{B}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) = \mathbf{C}\boldsymbol{x}(t) + \mathbf{D}\boldsymbol{u}(t) \end{cases} \tag{1.40}$$

where $\boldsymbol{u}, \boldsymbol{y} \in \mathbb{R}^P$ are the system input and output, respectively, and $\boldsymbol{x} \in \mathbb{R}^N$ are the system internal states.
The notation that we provide for the state-space realization is the following

$$\mathbf{H}(s) = \mathbf{D} + \mathbf{C}(s\mathbb{I} - \mathbf{A})^{-1}\mathbf{B} \leftrightarrow \left(\begin{array}{c|c} \mathbf{A} & \mathbf{B} \\ \hline \mathbf{C} & \mathbf{D} \end{array}\right) \tag{1.41}$$

Considering now the model of (1.34), with $\varphi(s)$ defined as the partial-fraction basis with a prescribed set of real and complex poles q_n (see Section 1.2.1), we can write

$$\mathsf{N}(s) = \mathbf{R}_0 + \sum_{n=1}^{\bar{n}} \frac{\mathbf{R}_n}{s - q_n} \tag{1.42}$$

$$\mathsf{D}(s) = r_0 + \sum_{n=1}^{\bar{n}} \frac{r_n}{s - q_n} \,. \tag{1.43}$$

We now construct the two separate state-space realizations as

$$\mathsf{N}(s) \leftrightarrow \left(\begin{array}{c|c} \mathbf{A}_0 & \mathbf{B}_0 \\ \hline \mathbf{C}_1(s) & \mathbf{D}_1(s) \end{array}\right) \tag{1.44}$$

$$\mathsf{D}(s)\mathbb{I}_P \leftrightarrow \left(\begin{array}{c|c} \mathbf{A}_0 & \mathbf{B}_0 \\ \hline \mathbf{C}_2(s) & \mathbf{D}_2(s) \end{array}\right) \,, \tag{1.45}$$

where

$$\mathbf{A}_0 = \text{blkdiag}\{\mathbf{A}_{0r}, \mathbf{A}_{0c}\} \tag{1.46}$$

$$\mathbf{B}_0^\mathsf{T} = \big[\mathbf{B}_{0r}^\mathsf{T}, \mathbf{B}_{0c}^\mathsf{T}\big] \tag{1.47}$$

$$\mathbf{C}_1 = \begin{bmatrix} \mathbf{R}_1 & \cdots & \mathbf{R}_{\bar{n}} \end{bmatrix} \tag{1.48}$$

$$\mathbf{C}_2 = \begin{bmatrix} \mathbb{I}_P\, r_1 & \cdots & \mathbb{I}_P\, r_{\bar{n}} \end{bmatrix} \tag{1.49}$$

$$\mathbf{D}_1 = \mathbf{R}_0 \tag{1.50}$$

$$\mathbf{D}_2 = \mathbb{I}_P\, r_0. \tag{1.51}$$

with

$$\mathbf{A}_{0r} = \text{blkdiag}\{q_n \mathbb{I}_P\}_{n=1}^{\bar{n}_r} \tag{1.52}$$

$$\mathbf{A}_{0c} = \text{blkdiag}\left\{ \begin{bmatrix} q'_n \mathbb{I}_P & q''_n \mathbb{I}_P \\ -q''_n \mathbb{I}_P & q'_n \mathbb{I}_P \end{bmatrix} \right\}_{n=1}^{\bar{n}_c} \tag{1.53}$$

$$\mathbf{B}_{0r} = \big[1,\ldots,1\big]^{\mathsf{T}} \otimes \mathbb{I}_P \tag{1.54}$$

$$\mathbf{B}_{0c} = \big[2,0,\ldots,2,0\big]^{\mathsf{T}} \otimes \mathbb{I}_P \tag{1.55}$$

where real-valued matrices have been used for complex conjugate poles.
We finally obtain the (compact) model state-space realization by the cascade of expression (1.44) as

$$\mathbf{H}(s) = \mathbf{N}(s)(\mathsf{D}(s)^{-1}\mathbb{I}_P) \leftrightarrow \left(\begin{array}{c|c} \mathbf{A}_0 - \mathbf{B}_0\mathbf{D}_2^{-1}\mathbf{C}_2 & \mathbf{B}_0\mathbf{D}_2^{-1} \\ \hline \mathbf{C}_1 - \mathbf{D}_1\mathbf{D}_2^{-1}\mathbf{C}_2 & \mathbf{D}_1\mathbf{D}_2^{-1} \end{array} \right) \tag{1.56}$$

We recall [19] and [24] for more details.

Descriptor Form

We now define an alternative descriptor form (or differential-algebraic system of equations (DAE), see [19]) to (1.56) as

$$\begin{cases} \mathbf{E}\dot{\boldsymbol{x}}(t) = \mathbf{A}\boldsymbol{x}(t) + \mathbf{B}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) = \mathbf{C}\boldsymbol{x}(t) \end{cases} \tag{1.57}$$

where $\boldsymbol{u}$ and $\boldsymbol{y}$ are the system input and output, respectively, and the system internal states are $\boldsymbol{x} \in \mathbb{R}^{N+P}$, with $N = \bar{n}P$: the number of states changes with respect to the state-space realization, increasing the problem dimension. The descriptor matrices of (1.57) are realized as

$$\mathbf{E} = \begin{pmatrix} \mathbb{I}_N & \mathbf{0}_{N,P} \\ \mathbf{0}_{P,N} & \mathbf{0}_{P,P} \end{pmatrix} \quad \mathbf{A} = \begin{pmatrix} \mathbf{A}_0 & \mathbf{B}_0 \\ \mathbf{C}_2 & \mathbf{D}_2 \end{pmatrix}$$
$$\mathbf{C} = \begin{pmatrix} \mathbf{C}_1 & \mathbf{D}_1 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} \mathbf{0}_{N,P} \\ -\mathbb{I}_P \end{pmatrix} \tag{1.58}$$

with $\mathbf{0}_{J,K}$ null matrix of size $J \times K$. The other matrices of (1.58) denote the state-space realization of the model numerator $\mathbf{N}(s)$, described by the set $\{\mathbf{A}_0, \mathbf{B}_0, \mathbf{C}_1, \mathbf{D}_1\}$, and the (extended) denominator $\mathsf{D}(s)\mathbb{I}_P$, described by $\{\mathbf{A}_0, \mathbf{B}_0, \mathbf{C}_2, \mathbf{D}_2\}$, which are exactly the same elements of (1.44).
It can be proven that the model expression of (1.34) is equivalent to

$$\mathbf{H}(s) = \mathbf{C}(s\mathbf{E} - \mathbf{A})^{-1}\mathbf{B} \tag{1.59}$$

as detailed in [19].
The descriptor form is particularly useful because it requires no block matrix inversion and moreover all matrix elements depend linearly on the model coefficients, in opposition with the regular state space realization of (1.34).
In the following sections we are going to describe in more details how the model should reflect the physical properties of the true system.

1.5 Stability

Several stability definitions may be formulated for an LTI system, analysing the general properties of all the possible solutions of a system. During our work we only modelled black-box systems, which can be characterized, from a stability standpoint, through the matrix $\mathbf{A}$ of the state-space realization (1.40) of the model $\mathbf{H}(s)$.
For this reason, we can define an LTI system [23] [30] [42] as

- *asymptotically stable* if and only if all the poles have a strictly negative real part, $\mathrm{Re}\{q_n\} < 0 \; \forall n$;
- *stable* if and only if all the poles have a non-positive real part, $\mathrm{Re}\{q_n\} \leq 0 \; \forall n$, and all the purely imaginary poles have a multiplicity that is at most one;
- *unstable* if at least one pole has either a strictly positive real part $\mathrm{Re}\{q_n\} > 0$ or a null real part with a multiplicity higher than one.

Furthermore, since the eigenvalues of $\mathbf{A}$ are the model poles q_n from (1.36), the matrix $\mathbf{A}$ can be denoted as (*asymptotically*) *stable* if its eigenvalues have a (strictly) negative real part.

1.6 Passivity

In electronic systems engineering, it is a common practice to deal with many interconnected sub-systems. Especially during high-speed electronic devices design, it is fundamental to assess the signal and power integrity (SI, PI), when all the sub-systems are connected together, since even individual components like vias and packages may strongly affect SI and PI performances if the design is poor. In general, it is common to perform in-depth analyses of these components and, to speed-up the whole process, surrogate macro-models for each sub-system are used, that will be connected together just in simulation phases. Such analyses of interconnected systems may suffer from instabilities, even if all the models are internally asymptotically stable. In fact, if one or more of the single macro-models is not passive, an un-physical energy generation may occur, leading to a distorted output signal which may have detrimental effects on the whole system. This fact, under suitable load conditions, may be responsible of an uncontrolled amplification of the output signal, resulting in an unstable simulation.
Model passivity turns out to be a fundamental requirement that must be carefully analysed when such macro-models are synthesized to ensure reliable simulations under any working condition.

The passivity of a system is strongly related to the net power it absorbs at any time instant t. Considering a P-ports system, the absorbed instantaneous power is

$$p(t) = \sum_{k=1}^{P} p_k(t) = \sum_{k=1}^{P} v_k(t) i_k(t) \tag{1.60}$$

that can be written in compact form as

$$p(t) = \mathbf{v}(t)^{\mathsf{T}} \mathbf{i}(t) = \mathbf{i}(t)^{\mathsf{T}} \mathbf{v}(t) \tag{1.61}$$

where $\mathbf{v} = [v_1, \ldots, v_k]^\mathsf{T}$ and $\mathbf{i} = [i_1, \ldots, i_k]^\mathsf{T}$.

In case the system is in immittance representation, input and output variables, denoted respectively as $u_k(t)$ and $y_k(t)$ may be either voltage or currents. The instantaneous power is thus

$$p(t) = \mathbf{y}^\mathsf{T}(t)\mathbf{u}(t) = \mathbf{u}^\mathsf{T}(t)\mathbf{y}(t) \tag{1.62}$$

Considering input and output as complex valued signals, the instantaneous power definition can be generalized, as

$$p(t) = \mathrm{Re}\left\{\mathbf{v}^\mathsf{H}(t)\mathbf{i}(t)\right\} = \mathrm{Re}\left\{\mathbf{i}^\mathsf{H}(t)\mathbf{v}(t)\right\} \tag{1.63}$$

For scattering representations, voltages v_k and currents i_k are transformed in incident and reflected scattering waves, respectively a_k and b_k. To this end we recall that

$$a_k = \frac{1}{2\sqrt{R_{ref,k}}}(v_k + R_{ref,k} i_k) \tag{1.64}$$

$$b_k = \frac{1}{2\sqrt{R_{ref,k}}}(v_k - R_{ref,k} i_k) \tag{1.65}$$

where $R_{ref} > 0$ is the normalization resistance of each port.
The power $p(t)$ for scattering representation is thus

$$p(t) = \sum_{k=1}^{P} \sqrt{R_{ref,k}}[a_k(t) + b_k(t)] \frac{1}{\sqrt{R_{ref,k}}}[a_k(t) - b_k(t)] = \mathbf{a}(t)^\mathsf{T}\mathbf{a}(t) - \mathbf{b}(t)^\mathsf{T}\mathbf{b}(t) \tag{1.66}$$

with $\mathbf{a}(t) = [a_1(t), \ldots, a_k(t)]$ and $\mathbf{b}(t) = [b_1(t), \ldots, b_k(t)]$.
Defining generic input and output signals as $\mathbf{u}(t) = \mathbf{a}(t)$ and $\mathbf{y}(t) = \mathbf{b}(t)$, it follows that

$$p(t) = \mathbf{u}(t)^\mathsf{T}\mathbf{u}(t) - \mathbf{y}(t)^\mathsf{T}\mathbf{y}(t) \tag{1.67}$$

Generalizing this definition to the case in which $\mathbf{u}$ and $\mathbf{y}$ are complex-valued signals, the instantaneous power is

$$p(t) = \mathbf{u}(t)^\mathsf{H}\mathbf{u}(t) - \mathbf{y}(t)^\mathsf{H}\mathbf{y}(t) \tag{1.68}$$

The net energy absorbed by a P-ports system in a time interval $[t_1,\ t_2]$ is defined as

$$\mathcal{E}(t) = \int_{t_1}^{t_2} p(\tau) d\tau \tag{1.69}$$

If the energy for $t_1 \to -\infty$ is vanishing, the cumulative net energy at an arbitrary time instant t is

$$\mathcal{E}(t) = \int_{-\infty}^{t} p(\tau) d\tau \tag{1.70}$$

The definition for passivity now can be stated.

Definition 1.1 *[19, 42, 43] A P-ports system is passive if the cumulative net energy in* (1.70) *is non-negative for any time t*

$$\mathcal{E}(t) \geq 0, \ \forall t \tag{1.71}$$

and for any input signal $\mathbf{u}(t)$.

The term "passivity" is often replaced by its synonym "dissipativity", so that a passive system is also denoted as "dissipative".

1.6.1 The Dissipation Inequality

The passivity definition given in the previous section regards only the net input/output energy flow, without making any reference to the system internal energy. An alternative way to describe the passivity of a system is to relate the amount of energy it stores and exchanges with the environment, for any time t. Considering a generic system (described in its state space representation) the following dissipativity definition holds:

Definition 1.2 *[19] A system (expressed in its state space representation) is dissipative with respect to the supply rate* $p(t)$ *if there exist a scalar-valued function* $V(\mathbf{x})$*, with* $\mathbf{x}$ *the system states, such that*

$$V(\mathbf{x}(t_1)) \leq V(\mathbf{x}(t_0)) + \int_{t_0}^{t_1} p(t)dt, \ \forall \ t_0 \leq t_1 \, and \, \forall \ \mathbf{u}, \mathbf{y}, \mathbf{x}. \tag{1.72}$$

.
The integral term in (1.72) is exactly the net cumulative energy entering the system in the time interval $[t_0, \ t_1]$, as defined in (1.69). The function $V(\mathbf{x}(t))$ is recognized to be the energy that is stored by the system at any time instant t. Equation (1.72) states that in a system, to be dissipative, the variation on internal energy $V(\mathbf{x}(t_1)) - V(\mathbf{x}(t_0))$ can not exceed the energy that is supplied from the environment to the system during the time interval $[t_0, \ t_1]$.
If the storage function is differentiable, Equation (1.72) can be rewritten in differential form as

$$\frac{d}{dt}V(\mathbf{x}(t)) \leq p(t), \ \forall t \tag{1.73}$$

Under the assumption that the energy stored for $t \to -\infty$ is vanishing, inequality (1.72) reduces to the passivity condition in Equation (1.71). This way to characterize the passivity of a system will turn out to be useful later on, when advanced algebraic passivity assessment methods will be derived

1.6.2 Passivity Characterization

Considering now the class of MIMO (Multi Input-Multi Output) lumped LTI systems with input $\mathbf{u}(t)$ and output $\mathbf{y}(t)$, for which there exist a transfer matrix representation,

the previous dissipativity definition can be written in terms of the transfer function, denoted as $\mathbf{H}(s)$, for both immittance and scattering representations.
For an immittance system, in order to derive passivity conditions in terms of its transfer matrix $\mathbf{H}(s)$, we can explicitly write the instantaneous absorbed power under cisoidal excitation $\mathbf{u}(s) = \mathbf{u}\, e^{st}$ using (1.63) as

$$p(t) = \mathrm{Re}\left\{\mathbf{u}^{\mathsf{H}}\mathbf{H}\mathbf{u}\right\} e^{2\sigma t},\ \sigma = \mathrm{Re}\{s\} \tag{1.74}$$

The cumulative net energy can be computed as

$$\mathcal{E}(t) = \int_{-\infty}^{t} p(\tau)d\tau = \mathrm{Re}\left\{\mathbf{u}^{\mathsf{H}}\mathbf{H}(s)\mathbf{u}\right\} \frac{e^{2\sigma t}}{2\sigma} \tag{1.75}$$

where $\sigma > 0$ to ensure the integral convergence.
Recalling the passivity condition in (1.71), it must hold $\mathcal{E}(t) \geq 0,\ \forall t$. Thus, being $\frac{e^{2\sigma t}}{2\sigma} > 0$ by assumption, it follows that

$$\mathrm{Re}\left\{\mathbf{u}^{\mathsf{H}}\mathbf{H}(s)\mathbf{u}\right\} = \mathbf{u}^{\mathsf{H}}\left[\frac{1}{2}(\mathbf{H}(s) + \mathbf{H}^{\mathsf{H}}(s))\right]\mathbf{u} \geq 0,\ \forall \mathbf{u} \in \mathbb{C}^{P} \tag{1.76}$$

We can conclude that an immittance system is dissipative if

$$\mathbf{H}(s) + \mathbf{H}^{\mathsf{H}}(s) \geq 0,\ \mathrm{Re}\{s\} > 0. \tag{1.77}$$

For further details on these derivations see [19].

To derive passivity conditions for scattering systems, as for the immittance case, the instantaneous power is written in terms of $\mathbf{H}(s)$. Under cisoidal excitation $\mathbf{u}(t)$, recalling Equation (1.68), it reads

$$p(t) = \mathbf{u}(t)^{\mathsf{H}}\mathbf{u}(t) - \mathbf{y}(t)^{\mathsf{H}}\mathbf{y}(t) = \mathbf{u}^{\mathsf{H}}[\mathbb{I} - \mathbf{H}(s)^{\mathsf{H}}\mathbf{H}(s)]\mathbf{u}\, e^{2\sigma t}. \tag{1.78}$$

As for the immittance case, we compute the cumulative net energy absorbed by the system at time instant t as

$$\mathcal{E}(t) = \int_{-\infty}^{t} p(\tau)d\tau = \mathbf{u}^{\mathsf{H}}\left[\mathbb{I} - \mathbf{H}^{\mathsf{H}}(s)\mathbf{H}(s)\right]\mathbf{u}\frac{e^{2\sigma t}}{2\sigma} \tag{1.79}$$

with $\sigma > 0$. The passivity condition in (1.71) implies that

$$\mathbb{I} - \mathbf{H}(s)^{\mathsf{H}}\mathbf{H}(s) \geq 0,\ \mathrm{Re}\{s\} > 0. \tag{1.80}$$

The two passivity conditions for immittance and scattering representation given above are now generalized with reference to Positive Real and Bounded Real matrices [4,19,41].

Definition 1.3 *A transfer matrix* $\mathbf{H}(s)$ *is Positive Real if:*

1. *each element of* $\mathbf{H}(s)$ *is defined and analytic in* $\mathrm{Re}\{s\} > 0$

2. $\mathbf{H}^*(s) = \mathbf{H}(s^*)$
3. $\mathbf{\Theta}(s) = \mathbf{H}(s) + \mathbf{H}^{\mathsf{H}}(s) \geq 0$ *for* $\mathrm{Re}\{s\} > 0$

Definition 1.4 *A transfer matrix* $\mathbf{H}(s)$ *is Bounded Real if:*

1. *each element of* $\mathbf{H}(s)$ *is defined and analytic in* $\mathrm{Re}\{s\} > 0$
2. $\mathbf{H}^*(s) = \mathbf{H}(s^*)$
3. $\mathbf{\Theta}(s) = \mathbb{I} - \mathbf{H}^{\mathsf{H}}(s)\mathbf{H}(s) \geq 0$ *for* $\mathrm{Re}\{s\} > 0$

Condition 1 is related to stability and causality. In fact both causality and stability requires the transfer function to be analytic (must not have poles) in the closed right half complex plane.

Condition 2 may be interpreted as a "consistency" one, since it implies that the transfer matrix is real for any $s \in \mathbb{R}$. This condition strongly affects the residues of $\mathbf{H}(s)$: in fact, to be satisfied, they must be real, for real poles, or must appear in complex conjugate pairs, when corresponding to complex conjugate poles.

Finally, Condition 3 is exactly the one we derived above in Equations 1.77 and 1.80, related to the energy of the system described by $\mathbf{H}(s)$.

We are now ready to re-formulate LTI system passivity conditions in terms of Positive Real and Bounded Real matrices, as stated in Theorem 1.1 ([4, 19, 41])

Theorem 1.1 *A LTI system with transfer matrix* $\mathbf{H}(s)$ *is defined to be passive if and only if* $\mathbf{H}(s)$ *is Positive Real (for immittance representations) or Bounded Real (for scattering representations).*

Theorem 1.1 provides a powerful theoretical tool to check the passivity of an LTI system through its transfer matrix. However, verifying that the three conditions are concurrently fulfilled in the open complex plane, implies considerable computational efforts.
In the following, we derive some simpler conditions, based on the rational nature of the model underlying the transfer matrix $\mathbf{H}(s)$ to assess whether the model is passive, for both immittance and scattering representations.

Considering immittance systems, the following Theorem holds [4, 19, 41]

Theorem 1.2 *A rational matrix* $\mathbf{H}(s)$ *is Positive Real if and only if*

1. $\mathbf{H}(s)$ *has no poles in* $\mathbb{C}_+$
2. $\mathbf{H}^*(\mathrm{j}\omega) = \mathbf{H}(-\mathrm{j}\omega)$
3. $\mathbf{H}(\mathrm{j}\omega) + \mathbf{H}^{\mathsf{H}}(\mathrm{j}\omega) \geq 0,\ \forall \omega \in \mathbb{C}$*, except for simple poles* $\mathrm{j}\omega_i$ *of* $\mathbf{H}(s)$ *where the transfer matrix must be Hermitian and nonnegative definite.*

4. *for* $\omega \to \infty$, $\mathbf{H}(s) \sim \mathbf{R}_\infty s$ *in* $\mathrm{Re}\{s\} > 0$, *with* $\mathbf{R}_\infty$ *real, symmetric and non-negative definite*

The main advantage of this theorem with respect to the more general one, as shown in [19], is evident from the third condition. In fact, comparing it with the one defined in 1.3, it turns out that the non-negative definiteness of $\mathbf{H}(s) + \mathbf{H}^{\mathsf{H}}(s)$ can be checked just along the imaginary axis rather than in the right half open complex plane.
If Conditions 1,2,4 are satisfied (as usually are), the only thing we need to check is Condition 3, whose statement can be cast as follows

$$\lambda_{min}(\mathrm{j}\omega) \geq 0, \ \forall \omega \in \mathbb{R} \tag{1.81}$$

with

$$\lambda_{min}(\mathrm{j}\omega) = \min\{\lambda(\mathbf{H}(\mathrm{j}\omega) + \mathbf{H}^{\mathsf{H}}(\mathrm{j}\omega)\}, \ \forall \omega \in \mathbb{R} \tag{1.82}$$

Assuming the transfer matrix to be asymptotically stable, the above eigenvalues are continuous functions of frequency, therefore $\lambda_{min}(\mathrm{j}\omega)$ is a continuous function of frequency. This fact enables the use frequency sampling techniques in advanced passivity assessment algorithms.
In the next sections we will go through a set of fundamental results to perform advanced and reliable passivity verifications.

Immittance Systems

Particularizing now the dissipation inequality 1.73 for immittance LTI systems, we will derive a condition to assess system passivity in terms of the state-space representation matrices.
To this end, we need to have an analytic expression of the supplied power that is given by Equation 1.74 and reads

$$p(t) = \frac{1}{2}[\mathbf{u}^{\mathsf{T}}(\mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}) + (\mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u})^{\mathsf{T}}\mathbf{u}] \tag{1.83}$$

If the storage function is defined as $V(\mathbf{x}) = \frac{1}{2}(\mathbf{x}^{\mathsf{T}}\mathbf{P}\mathbf{x})$, with $\mathbf{P}$ a symmetric positive definite matrix, its derivative (rate of change of the internal energy) will be

$$\frac{d}{dt}V(\mathbf{x}(t)) = \frac{1}{2}[(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u})^{\mathsf{T}}\mathbf{P}\mathbf{x} + \mathbf{x}^{\mathsf{T}}\mathbf{P}(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u})] \tag{1.84}$$

Let us now impose the dissipativity condition defined in Equation 1.71. Splitting input and state signals, with trivial algebraic manipulations we get to the following LMI form, known as *Positive Real Lemma* [4, 34].

Lemma 1.1 *A LTI system in immittance form is passive if and only if, for any signal* $\mathbf{x}, \mathbf{u}$ *satisfying the state equations, it holds that:*

$$\exists \mathbf{P} = \mathbf{P}^{\mathsf{T}} > 0 : \quad \begin{pmatrix}\mathbf{x}\\ \mathbf{u}\end{pmatrix}^{\mathsf{T}} \begin{pmatrix}\mathbf{A}^{\mathsf{T}}\mathbf{P} + \mathbf{P}\mathbf{A} & \mathbf{P}^{\mathsf{T}}\mathbf{B} - \mathbf{C}^{\mathsf{T}} \\ \mathbf{B}^{\mathsf{T}}\mathbf{P} - \mathbf{C} & -(\mathbf{D} + \mathbf{D}^{\mathsf{T}})\end{pmatrix} \begin{pmatrix}\mathbf{x}\\ \mathbf{u}\end{pmatrix} \leq 0 \tag{1.85}$$

We now derive a fundamental result, originally proposed in [19], used extensively in LTI passivity assessment algorithms, that enables the use of algebraic methods to spot passivity violations. In details, it will be shown that the imaginary eigenvalues of a particular Hamiltonian-structured matrix are strongly related to the location of passivity violations along the frequency axis.
First, let us define a support matrix function, called *Popov function*, $\mathbf{\Psi}(s)$ as

$$\mathbf{\Psi}(s) = \mathbf{H}(s) + \mathbf{H}^{\mathsf{T}}(-s) \tag{1.86}$$

Recalling that, to be passive, the transfer matrix of an immittance system must satisfy

$$\mathbf{\Theta}(s) = \mathbf{H}(s) + \mathbf{H}^{\mathsf{H}}(s) \geq 0 \tag{1.87}$$

in turns out that $\mathbf{\Theta}(s)$ and $\mathbf{\Psi}(s)$ are equal when evaluated on the imaginary axis. This enables us to check the non-negative definiteness of $\mathbf{\Psi}(\mathrm{j}\omega)$ instead of $\mathbf{\Theta}(\mathrm{j}\omega)$.
The condition that must be verified to guarantee passivity is thus

$$\mathbf{\Psi}(\mathrm{j}\omega) \geq 0, \forall\omega \tag{1.88}$$

Focusing our attention to this last equation, we see that the frequencies at which $\mathbf{\Psi}(\mathrm{j}\omega_i)$ becomes singular, algebraically pinpoint passivity violations, being $\mathbf{\Psi}(\mathrm{j}\omega_i)$ singular exactly when $\mathbf{H}(\mathrm{j}\omega) + \mathbf{H}^{\mathsf{T}}(-\mathrm{j}\omega) = 0$.
These frequencies $\mathrm{j}\omega_i$ are defined to be the solutions of

$$\mathbf{\Psi}(\mathrm{j}\omega_i)\mathbf{u} = 0 \tag{1.89}$$

for some vector $\mathbf{u}$.
In order to find these frequencies, we derive a state-space realization of $\mathbf{\Psi}(s)$, the analytic extension to the open complex plane of $\mathbf{\Psi}(\mathrm{j}\omega)$. This turns out to be useful since the solutions of Equation (1.89) are the poles of $\mathbf{\Psi}^{-1}(s)$, for which a simple state space realization is readily computed. The poles of $\mathbf{\Psi}^{-1}(s)$ are the eigenvalues of its dynamic matrix, that reads

$$\boldsymbol{\mathcal{N}}_0 = \mathbf{A}_{\mathbf{\Psi}^{-1}} = \mathbf{A}_\Psi - \mathbf{B}_\Psi \mathbf{D}_\Psi^{-1} \mathbf{C}_\Psi \tag{1.90}$$

where $\mathbf{A}_\Psi,\ \mathbf{B}_\Psi,\ \mathbf{C}_\Psi,\ \mathbf{D}_\Psi$ are the state-space realization matrices of $\mathbf{\Psi}(s)$.
Expanding $\boldsymbol{\mathcal{N}}_0$ in terms of the system realization matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$ we get the following matrix

$$\boldsymbol{\mathcal{N}}_0 = \begin{pmatrix} \mathbf{A} - \mathbf{B}(\mathbf{D} + \mathbf{D}^{\mathsf{T}})^{-1}\mathbf{C} & -\mathbf{B}(\mathbf{D} + \mathbf{D}^{\mathsf{T}})^{-1}\mathbf{B}^{\mathsf{T}} \\ \mathbf{C}^{\mathsf{T}}(\mathbf{D} + \mathbf{D}^{\mathsf{T}})^{-1}\mathbf{C} & -\mathbf{A}^{\mathsf{T}} + \mathbf{C}^{\mathsf{T}}(\mathbf{D} + \mathbf{D}^{\mathsf{T}})^{-1}\mathbf{B}^{\mathsf{T}} \end{pmatrix} \tag{1.91}$$

Defining as $\mathbf{J}$ the following matrix

$$\mathbf{J} = \begin{pmatrix} 0 & \mathbb{I}_n \\ -\mathbb{I}_n & 0 \end{pmatrix} \tag{1.92}$$

it holds that

$$(\mathbf{J}\boldsymbol{\mathcal{N}}_0)^{\mathsf{T}} = \mathbf{J}\boldsymbol{\mathcal{N}}_0 \tag{1.93}$$

which shows that $\boldsymbol{\mathcal{N}}_0$ has a Hamiltonian structure.
Because of that, $\boldsymbol{\mathcal{N}}_0$ has some peculiar characteristics. In particular, its eigen-spectrum is symmetric with respect to both imaginary and real axes. In fact the set of poles of $\boldsymbol{\Psi}(s)$ includes the ones of $\mathbf{H}(s)$ which are symmetric with respect to the real axis, and their mirror images, symmetric with respect to the imaginary axis.
The following theorem, proposed in [6,15,19], provides a fundamental results that relates the eigenvalues of $\boldsymbol{\mathcal{N}}_0$ with the ones of $\boldsymbol{\Psi}(\mathrm{j}\omega)$.

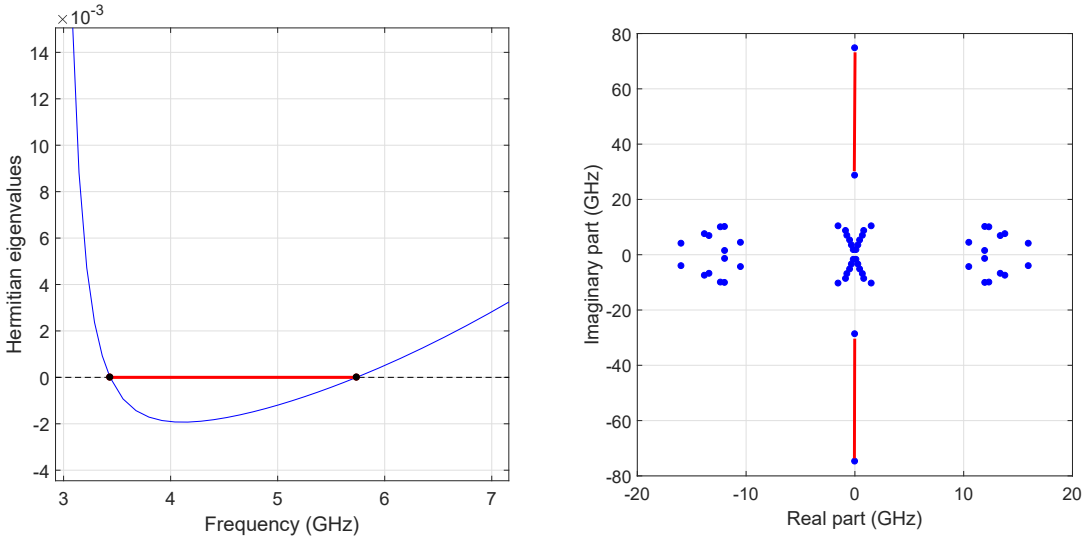
Theorem 1.3 *Let $\mathbf{H}(s)$ be the transfer matrix of an immittance system, whose state space matrices are $\{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}\}$, where $\mathbf{A}$ has no purely imaginary eigenvalues and $\mathbf{D}+\mathbf{D}^{\mathsf{T}}$ is non-singular. Then, $\mathrm{j}\omega_0$ is an eigenvalue of $\boldsymbol{\mathcal{N}}_0$ if and only if 0 is an eigenvalue of $\boldsymbol{\Psi}(\mathrm{j}\omega_0)$.*

It follows that, if $\boldsymbol{\mathcal{N}}_0$ has imaginary eigenvalues, the related LTI system is not passive for some frequency bands.
This result is formally stated in Theorem 1.4.

Theorem 1.4 *Let $\mathbf{H}(s)$ be the transfer matrix of an immittance system, whose state space matrices are $\{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}\}$, where $\mathbf{A}$ has no purely imaginary eigenvalues and $\mathbf{D}+\mathbf{D}^{\mathsf{T}}$ is positive definite. Then the system is passive if the Hamiltonian matrix $\boldsymbol{\mathcal{N}}_0$ has no purely imaginary eigenvalues.*

Theorems 1.3 and 1.4 provide an algebraic tool that is able to precisely verify system passivity and enables us to easily localize violation areas along the frequency axis.
To this end we must notice that Hamiltonian imaginary eigenvalues correspond to the complex frequencies at which at least one eigenvalue of $\boldsymbol{\Psi}(\mathrm{j}\omega)$ crosses the zero-threshold. These frequencies induce a partition of the frequency axis in disjoint sub-bands, where $\boldsymbol{\Psi}(\mathrm{j}\omega)$ is either positive definite or not. This means that, being the Hamiltonian eigenvalues the edges of these sub-bands, the frequency axis is now partitioned in passive and not-passive areas, so that a detailed passivity characterization is available.
In Figure 1.1 we show the described partitioning of the frequency axis in passive and non-passive bands induced by imaginary Hamiltonian eigenvalues. In the left panel we show an eigenvalue of $\mathbf{H}(\mathrm{j}\omega)+\mathbf{H}^{\mathsf{H}}(\mathrm{j}\omega)$ that, becoming negative, denote a non-passive frequency band, shown in red. Imaginary Hamiltonian eigenvalues are represented as black dots and bound this violation area. In the right panel we show the Hamiltonian eigen-spectrum, where it is possible to see that the magnitude of purely imaginary eigenvalues coincides with the edges of the violation interval discussed before. The violation bands in the complex plane are represented with red lines.

The main result presented here relies on the strong assumption that $\mathbf{D}+\mathbf{D}^{\mathsf{T}}$ is not singular. However, the same approach can be extended to the case in which $\mathbf{D}+\mathbf{D}^{\mathsf{T}}$ is singular with minor modifications. For details see [19].
In order to relax the non-singularity condition on $\mathbf{D}+\mathbf{D}^{\mathsf{T}}$, it is necessary to slightly modify Theorem 1.3 resulting in an extended eigenvalue problem where, now, no inversions on $\mathbf{D}+\mathbf{D}^{\mathsf{T}}$ are required. The new problem, shown in Equation (1.94) is cast in what is usually called a "generalized eigenvalue problem", where the unknowns are no more the eigenvalues of a matrix, but the ones of a matrix pencil $(\boldsymbol{\mathcal{N}}_0^{ext},\boldsymbol{\mathcal{K}})$.



(a) Passive/non-passive frequency axis partitioning

(b) Violation bands in the Hamiltonian eigenspectrum

Figure 1.1: Frequency axis partitioning induced by Hamiltonian imaginary eigenvalues

$$\boldsymbol{\mathcal{N}}_0^{ext}\mathbf{v} = \mathrm{j}\omega_0\boldsymbol{\mathcal{K}}\mathbf{v} \tag{1.94}$$

where

$$\boldsymbol{\mathcal{N}}_0^{ext} = \begin{pmatrix} \mathbf{A} & \mathbf{0} & \mathbf{B} \\ \mathbf{0} & -\mathbf{A}^\mathsf{T} & -\mathbf{C}^\mathsf{T} \\ \mathbf{C} & \mathbf{B}^\mathsf{T} & \mathbf{D}+\mathbf{D}^\mathsf{T} \end{pmatrix}, \qquad \boldsymbol{\mathcal{K}} = \begin{pmatrix} \mathbb{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbb{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{1.95}$$

This matrix pencil is denoted as "*Skew-Hamiltonian/Hamiltonian*", because $\boldsymbol{\mathcal{N}}_0^{ext}$ has Hamiltonian structure while $\boldsymbol{\mathcal{K}}$ is skew-Hamiltonian.

Up to now, just a state-space realization for $\mathbf{H}(s)$ has been considered. However there are several situations for which a descriptor realization is preferable, e.g., when using MNA (Modified Nodal Analysis) method to automatically solve electrical circuits. For this reason, a generalization of the Hamiltonian approach to descriptor realization is needed. Minor modifications to Theorem 1.3 allow to state that, for immittance systems in descriptor form, the complex frequencies at which passivity violations occur are the purely imaginary generalized eigenvalues of this generalized eigen-problem:

$$\boldsymbol{\mathcal{N}}_0^{ext}\mathbf{v} = \mathrm{j}\omega_0\boldsymbol{\mathcal{K}}\mathbf{v} \tag{1.96}$$

where

$$\boldsymbol{\mathcal{N}}_0^{ext} = \begin{pmatrix} \mathbf{A} & \mathbf{0} & \mathbf{B} \\ \mathbf{0} & -\mathbf{A}^\mathsf{T} & -\mathbf{C}^\mathsf{T} \\ \mathbf{C} & \mathbf{B}^\mathsf{T} & \mathbf{D}+\mathbf{D}^\mathsf{T} \end{pmatrix}, \qquad \boldsymbol{\mathcal{K}} = \begin{pmatrix} \mathbf{E} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}^\mathsf{T} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{1.97}$$

Scattering systems

We now focus on scattering systems.
Recalling Theorem 1.1, a scattering system, to be passive, must have a Bounded Real transfer matrix. Again, verifying system passivity throughout the complex plane is too expensive in terms of computational effort.
As for the Positive Real Lemma, a formulation of the Bounded Real Lemma exists for rational matrices [4, 19, 41], that are the main focus of this work.

Theorem 1.5 *A rational matrix* $\mathbf{H}(s)$ *is Bounded Real if and only if*

1. $\mathbf{H}(s)$ *has no poles in* $\mathbb{C}_+$
2. $\mathbf{H}^*(\mathrm{j}\omega) = \mathbf{H}(-\mathrm{j}\omega)$
3. $\mathbb{I} - \mathbf{H}(\mathrm{j}\omega)^{\mathsf{H}}\mathbf{H}(\mathrm{j}\omega) \geq 0, \ \forall\omega \in \mathbb{R}$

No further conditions are required, as in the immittance case, for purely imaginary poles, because passive scattering systems can not have poles on the imaginary axis. As in Theorem 1.2, the main advantage that the rational nature of the system brings with it, is that Conditions 2 and 3 can be checked just along the imaginary axis. Assuming the system to be asymptotically stable (all the poles of $\mathbf{H}(s)$ has strictly negative real part) and that the state-space realization matrices real, the first two conditions are verified and only the third remains to be checked.
Here, in contrast with the immittance case, a product of transfer matrices appears, so a direct eigenvalues calculation, to guarantee that the smaller one is above the zero threshold, should be avoided. An alternative formulation for Condition 3 is based on the SVD (Singular Values Decomposition) of $\mathbf{H}(\mathrm{j}\omega)$, that reads

$$\mathbf{H}(\mathrm{j}\omega) = \mathbf{U}(\mathrm{j}\omega)\mathbf{\Sigma}(\mathrm{j}\omega)\mathbf{V}(\mathrm{j}\omega)^{\mathsf{H}} \tag{1.98}$$

The third condition is then re-formulated in terms of the singular values of $\mathbf{H}(\mathrm{j}\omega)$:

$$\mathbb{I} - \mathbf{H}(\mathrm{j}\omega)^{\mathsf{H}}\mathbf{H}(\mathrm{j}\omega) \geq 0 \Leftrightarrow \sigma_{max}(\mathbf{H}(\mathrm{j}\omega)) = \|\mathbf{H}(\mathrm{j}\omega)\|_2 \leq 1, \ \forall\omega \in \mathbb{R}. \tag{1.99}$$

Being additionally, by assumption, the transfer matrix $\mathbf{H}(\mathrm{j}\omega)$ regular in an open subset of the complex plane containing the imaginary axis, singular values are continuous functions of $\mathrm{j}\omega$, enabling the use of frequency sampling techniques.
Since any passive system must satisfy the dissipation inequality in (1.73), to derive a precise passivity characterization, it must be particularized for scattering systems.
The supplied power $p(t)$ is

$$p(t) = \mathbf{u}^{\mathsf{T}}\mathbf{u} - \mathbf{y}^{\mathsf{T}}\mathbf{y} = \mathbf{u}^{\mathsf{T}}\mathbf{u} - (\mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u})^{\mathsf{T}}(\mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}), \tag{1.100}$$

where $\mathbf{u}, \mathbf{y}$ are respectively the input and output signals and the time dependency has been omitted for readability.
The storage function $V(\mathbf{x})$, defined as $V(\mathbf{x}) = \mathbf{x}^{\mathsf{T}}\mathbf{P}\mathbf{x}$, with $\mathbf{P} = \mathbf{P}^{\mathsf{T}} \geq 0$, leads to the

following equation

$$\frac{d}{dt}V(\mathbf{x}(t)) = (\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{u})^{\mathsf{T}}\mathbf{P}\mathbf{x} + \mathbf{x}^{\mathsf{T}}\mathbf{P}(\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{u}) \leq p(t),\ \forall t. \tag{1.101}$$

Combining the previous relation with the dissipation inequality, and splitting the input and state signals, the so-called *Bounded Real Lemma* [4,34] can be stated.

Lemma 1.2 *A LTI system in scattering form is passive if and only if, for any signal* $\mathbf{x},\mathbf{u}$ *satisfying the state equations, it holds that:*

$$\exists \mathbf{P}=\mathbf{P}^{\mathsf{T}}>0: \quad \begin{pmatrix}\mathbf{x}\\ \mathbf{u}\end{pmatrix}^{\mathsf{T}} \begin{pmatrix}\mathbf{A}^{\mathsf{T}}\mathbf{P}+\mathbf{P}\mathbf{A}+\mathbf{C}^{\mathsf{T}}\mathbf{C} & \mathbf{P}\mathbf{B}+\mathbf{C}^{\mathsf{T}}\mathbf{D}\\ \mathbf{B}^{\mathsf{T}}\mathbf{P}+\mathbf{D}^{\mathsf{T}}\mathbf{C} & -(\mathbb{I}-\mathbf{D}^{\mathsf{T}}\mathbf{D})\end{pmatrix}\begin{pmatrix}\mathbf{x}\\ \mathbf{u}\end{pmatrix} \leq 0 \tag{1.102}$$

In the following we derive, as for immittace representations, a set of results that enables to cast the passivity verification problem in a closed algebraic form. See [19] for details.
Defining $\boldsymbol{\Theta}(s)$ as

$$\boldsymbol{\Theta}(s) = \mathbb{I} - \mathbf{H}^{\mathsf{H}}(s)\mathbf{H}(s), \tag{1.103}$$

and denoting the Popov function as

$$\boldsymbol{\Psi}(s) = \mathbb{I} - \mathbf{H}^{\mathsf{T}}(-s)\mathbf{H}(s), \tag{1.104}$$

it is easy to see that, when evaluating these functions for $s=\mathrm{j}\omega$, they are equal:

$$\boldsymbol{\Psi}(\mathrm{j}\omega) = \boldsymbol{\Theta}(\mathrm{j}\omega). \tag{1.105}$$

Passivity condition can be cast in terms of the Popov function as

$$\boldsymbol{\Psi}(\mathrm{j}\omega) \geq 0,\ \forall\omega \tag{1.106}$$

Equation 1.106 exactly matches the one for immittance representations, where passivity violations are solutions of

$$\boldsymbol{\Psi}(\mathrm{j}\omega)\mathbf{u} = 0 \tag{1.107}$$

for some vector $\mathbf{u}$.
To find the zeros of $\boldsymbol{\Psi}(\mathrm{j}\omega)$, a state space realization for $\boldsymbol{\Psi}(s)$ (whose matrices are $\mathbf{A}_\Psi,\mathbf{B}_\Psi,\mathbf{C}_\Psi,\mathbf{D}_\Psi$) is derived, from which it is possible to get a realization for $\boldsymbol{\Psi}^{-1}(s)$, whose purely imaginary poles are the solutions of Equation (1.107). The poles of $\boldsymbol{\Psi}^{-1}(s)$ are the eigenvalues of its state-space dynamic matrix, that reads:

$$\boldsymbol{\mathcal{M}}_1 = \mathbf{A}_\Psi - \mathbf{B}_\Psi\mathbf{D}_\Psi^{-1}\mathbf{C}_\Psi. \tag{1.108}$$

Writing now this matrix in terms the state-space realization matrices $\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}$ of $\mathbf{H}(s)$, we get the following matrix:

$$\boldsymbol{\mathcal{M}}_1 = \begin{pmatrix}\mathbf{A}+\mathbf{B}(\mathbb{I}-\mathbf{D}^{\mathsf{T}}\mathbf{D})^{-1}\mathbf{D}^{\mathsf{T}}\mathbf{C} & \mathbf{B}(\mathbb{I}-\mathbf{D}^{\mathsf{T}}\mathbf{D})^{-1}\mathbf{B}^{\mathsf{T}}\\ -\mathbf{C}^{\mathsf{T}}(\mathbb{I}-\mathbf{D}\mathbf{D}^{\mathsf{T}})^{-1}\mathbf{C} & -\mathbf{A}^{\mathsf{T}}-\mathbf{C}^{\mathsf{T}}\mathbf{D}(\mathbb{I}-\mathbf{D}^{\mathsf{T}}\mathbf{D})^{-1}\mathbf{B}^{\mathsf{T}}\end{pmatrix} \tag{1.109}$$

Matrix $\boldsymbol{\mathcal{M}}_1$ has Hamiltonian structure, since it satisfies the condition in (1.93).

What relates matrix $\boldsymbol{\mathcal{M}}_1$ with system passivity is given by the following theorem [6,15,19]:

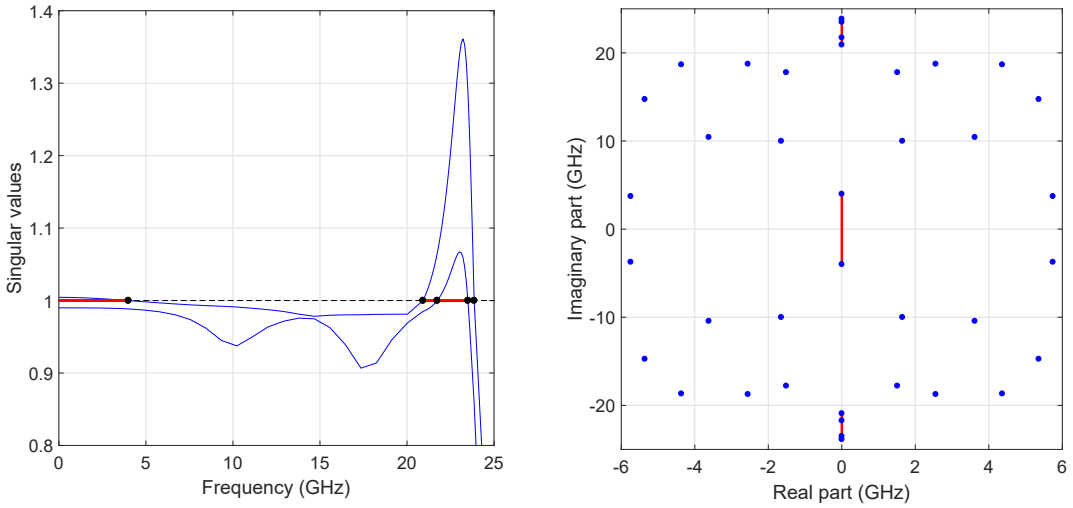
Theorem 1.6 *Let* $\mathbf{H}(s)$ *be the transfer matrix of a scattering system, whose state space matrices are* $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$*, where* $\mathbf{A}$ *has no purely imaginary eigenvalues and* $\mathbb{I} - \mathbf{D}^\mathsf{T}\mathbf{D}$ *is non-singular. Then,* $\mathrm{j}\omega_0$ *is an eigenvalue of* $\boldsymbol{\mathcal{M}}_1$ *if and only if 0 is an eigenvalue of* $\boldsymbol{\Psi}(\mathrm{j}\omega_0)$ *and 1 a singular value of* $\mathbf{H}(\mathrm{j}\omega_0)$*.*

This result allows us to derive the following theorem, that provides a sufficient passivity condition for scattering systems:

Theorem 1.7 *Let* $\mathbf{H}(s)$ *be the transfer function of an asymptotically passive (*$\|\mathbf{D}\|_2 < 1$*) and stable scattering system, whose state-space matrices are* $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$*. The system is uniformly passive if* $\boldsymbol{\mathcal{M}}_1$ *has no purely imaginary eigenvalues*

Furthermore, the frequencies ω_i solving $\boldsymbol{\Psi}(\mathrm{j}\omega_i)\mathbf{u} = 0$, i.e., the Hamiltonian imaginary eigenvalues, induce a partition of the frequency axis in passive and not-passive sub-bands. These considerations allow to characterize in details the passivity of a system for any frequency value.

Figure 1.1 shows the partitioning of the frequency axis in passive and non-passive bands induced by imaginary Hamiltonian eigenvalues. In the left panel we show singular values of $\mathbf{H}(\mathrm{j}\omega)$ that, denote non-passive areas when exceed the unit threshold, represented in red. Imaginary Hamiltonian eigenvalues are represented as black dots and bound these violation areas. In the right panel we show the Hamiltonian eigen-spectrum, where we can see that the magnitude of purely imaginary eigenvalues coincide with the edges of the violations interval discussed before. The violations band in the complex plane are represented with red lines.



(a) Passive/non-passive frequency axis partitioning

(b) Violation bands in the Hamiltonian eigen-spectrum

Figure 1.2: Frequency axis partitioning induced by Hamiltonian imaginary eigenvalues

As we did for immittance systems, it is possible to relax the non-singularity condition on $\mathbb{I} - \mathbf{D}^\mathsf{T}\mathbf{D}$.

As proposed in [19], Theorem 1.6 can be generalized to the case in which $\mathbf{D}$ is arbitrary. Slightly modifying its proof, it is possible to define an extended eigen-problem shown in (1.110), that does not require any inversion of $\mathbb{I} - \mathbf{D}\mathbf{D}^\mathsf{T}$ and $\mathbb{I} - \mathbf{D}^\mathsf{T}\mathbf{D}$, as:

$$\boldsymbol{\mathcal{M}}_1^{ext}\mathbf{v} = \mathrm{j}\omega_0\boldsymbol{\mathcal{K}}\mathbf{v}, \tag{1.110}$$

where

$$\boldsymbol{\mathcal{N}}_0^{ext} = \begin{pmatrix} \mathbf{A} & \mathbf{0} & \mathbf{B} & \mathbf{0} \\ \mathbf{0} & -\mathbf{A}^\mathsf{T} & \mathbf{0} & -\mathbf{C}^\mathsf{T} \\ \mathbf{0} & \mathbf{B}^\mathsf{T} & -\mathbb{I} & \mathbf{D}^\mathsf{T} \\ \mathbf{C} & \mathbf{0} & \mathbf{D} & \mathbb{I} \end{pmatrix}, \qquad \boldsymbol{\mathcal{K}} = \begin{pmatrix} \mathbb{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbb{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{1.111}$$

It can be proven that purely imaginary eigenvalues of 1.110 correspond exactly to the location on the frequency axis of passivity violations.

Previous results are based on the assumption that a state-space realization for $\mathbf{H}(s)$ is used. Here, we provide a generalization of the Hamiltonian-driven passivity characterization to descriptor realizations, that will be used extensively later on in this work, and are of paramount importance in many other applications.
Passivity violations are again defined by complex frequencies $\mathrm{j}\omega_i$ for which $\boldsymbol{\Psi}(\mathrm{j}\omega_i)\mathbf{v} = 0$. Suitably modifying Theorem 1.6, we find that this condition is reached if and only if $\mathrm{j}\omega_i$ is an eigenvalue of the generalized eigenproblem in (1.112)

$$\boldsymbol{\mathcal{M}}_1^{ext}\mathbf{v} = \mathrm{j}\omega_0\boldsymbol{\mathcal{K}}\mathbf{v} \tag{1.112}$$

where

$$\boldsymbol{\mathcal{N}}_0^{ext} = \begin{pmatrix} \mathbf{A} & \mathbf{0} & \mathbf{B} & \mathbf{0} \\ \mathbf{0} & -\mathbf{A}^\mathsf{T} & \mathbf{0} & -\mathbf{C}^\mathsf{T} \\ \mathbf{0} & \mathbf{B}^\mathsf{T} & -\mathbb{I} & \mathbf{D}^\mathsf{T} \\ \mathbf{C} & \mathbf{0} & \mathbf{D} & \mathbb{I} \end{pmatrix}, \qquad \boldsymbol{\mathcal{K}} = \begin{pmatrix} \mathbf{E} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}^\mathsf{T} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{1.113}$$

Chapter 2

Multivariate Macromodels

This Chapter is co-authored by T.Bradde, M. De Stefano and A. Zanco.

In the previous chapter, we assumed that the system under modeling is characterized by a fixed (yet unknown) physical structure. In many situations, however, this hypothesis is not the most suitable: some physical parameters of the system could be design objectives or could be intrinsically uncertain due to production process tolerances. A parametric macromodel is able to reproduce the system behaviour for all the possible values that the varying parameters assume within a prescribed range. This possibility proves to be extremely useful in many fields of the design process, from the optimization of the design variables, to the simulation of worst-case scenarios induced by the physical realization of the structure. Typical examples regard the role of temperature in electronic devices, the geometrical parameters of an interconnect, the linearization point of a non-linear device, and many more.
The construction flow of a parametric macromodel requires the knowledge of the input-output behavior for a discrete number of values within the range that the parameters can span; once those data are collected and processed, the interpolation algorithm returns a closed form description of the system within the entire range of variation.
In this case, the input-output data must be representative of the model behavior within all the range of values assumed by each parameter; in particular, consider the case in which the model is required to depend on a number ρ of design parameters. Then, for the i-th parameter we can denote its variation range as

$$\Theta_i = \left[\theta^i_{min}, \theta^i_{max}\right] \quad for \quad i = 1,2,..,\rho \quad . \tag{2.1}$$

Thus, the global parameter domain can be defined as:

$$\Theta = \Theta_1 \times \Theta_2 \times ..., \times \Theta_\rho \quad \subseteq \mathbb{R}^\rho. \tag{2.2}$$

A point in Θ is uniquely identified by its projections along the parameters axes. To keep the notation compact, this point is denoted as

$$\vartheta_{\mathbf{m}} = (\vartheta_{m_1}, \ldots, \vartheta_{m_\rho})^T \tag{2.3}$$

where $\mathbf{m}$ is a multi-index $\mathbf{m} = [m_1, \ldots, m_\rho]$.
To synthetize a parametric macromodel, a set of M points in the parameter domain Θ is

defined to be representative of the parametric system response; for each of these points, we collect K frequency samples of the transfer functions associated with the underlying system. The resulting dataset reads:

$$\breve{\mathbf{H}}_{k,m} = \breve{\mathbf{H}}(s_k, \theta_m) \quad for \quad k = 1,2,...,K \quad m = 1,2,...,M \quad , \tag{2.4}$$

If, as it is common, we collect data at real frequencies ω_k, our goal is to obtain a model:

$$\mathbf{H}(\mathrm{j}\omega, \theta) \approx \breve{\mathbf{H}}(\mathrm{j}\omega, \theta) \quad for \quad \theta \in \Theta, \quad \omega \in [\omega_{min}, \omega_{max}] \tag{2.5}$$

While the structure of an univariate model is supposed to be a rational function of the Laplace variable, we are free to cast the dependence of the model on the parameters in a larger set of possible structures: a variety of basis functions can be used to fit the data. The thesis project is particularly focused on the investigation of issues related to the construction of precise and reliable parametric macromodels, for which many open problems still exist.

2.1 Parametric Model Formulation

Approximating the true system response $\breve{\mathbf{H}}(s_k, \vartheta_k)$ in a suitable macromodel form is fundamental to include the curve fitting result in system-level simulations using standard circuit solver such as SPICE. Several mathematical structures are available: the identification algorithm efficiency, in frequency and time domain, is affected by this choice. Moreover, all the formulations may suffer from ill-conditioning depending on the parameter-dependent basis choice.
Therefore, considering a P-ports multivariate macromodel of a generic LTI system, we adopt the so-called *Parameterized Sanathanan-Koerner (PSK)* [37], [36], [12], [11], [21] form

$$\mathbf{H}(s;\vartheta) = \frac{\mathbf{N}(s,\vartheta)}{\mathsf{D}(s,\vartheta)} = \frac{\sum_{n=0}^{\bar{n}} \sum_{\ell=1}^{\bar{\ell}} \mathbf{R}_{n,\ell}\, \xi_\ell(\vartheta)\, \varphi_n(s)}{\sum_{n=0}^{\bar{n}} \sum_{\ell=1}^{\bar{\ell}} r_{n,\ell}\, \xi_\ell(\vartheta)\, \varphi_n(s)}. \tag{2.6}$$

We remark that the model numerator and denominator are constructed by linear combination of suitable basis functions: it is straightforward to prove that if the basis functions $\varphi_n(s)$ are rational, the model indicated in (2.6) is a rational function $\forall\vartheta$.
In particular, we denoted with $\bar{n}$ the frequency basis order and with $\bar{\ell}$ the cardinality of the parameter-dependent basis function. To maintain the notation compact, we define a multi-index $\ell = (\ell_1, \ldots, \ell_\rho)$, if $\rho > 1$.
Both the numerator and denominator coefficients are guaranteed real-valued: they are indicated with $\mathbf{R}_{n,\ell} \in \mathbb{R}^{P\times P}$ and $r_{n,\ell} \in \mathbb{R}$, respectively, in (2.6). We can simplify the model expression presented in (2.6), gathering the parameter information

$$\mathbf{H}(s;\vartheta) = \frac{\mathbf{N}(s,\vartheta)}{\mathsf{D}(s,\vartheta)} = \frac{\sum_{n=0}^{\bar{n}} \mathbf{R}_n(\vartheta)\, \varphi_n(s)}{\sum_{n=0}^{\bar{n}} r_n(\vartheta)\, \varphi_n(s)}, \tag{2.7}$$

where

$$\mathbf{R}_n(\vartheta) = \sum_{\ell_N=1}^{\bar{\ell}_N} \mathbf{R}_{n,\ell_N}\, \xi_{\ell_N}(\vartheta) \quad r_n(\vartheta) = \sum_{\ell_D=1}^{\bar{\ell}_D} r_{n,\ell_D}\, \xi_{\ell_D}(\vartheta) \tag{2.8}$$

are the numerator and denominator model coefficients, respectively.
Note that a different parameter-dependent basis order for numerator($\bar{\ell}_N$) and denominator($\bar{\ell}_D$) polynomials is possible, as specified in (2.8). Without loss of generality, in the following we will set $\bar{\ell}_N = \bar{\ell}_D = \bar{\ell}$.
The model structure presented before is completely general with respect to the input data set $\breve{\mathbf{H}}(s_k, \vartheta_k)$ representation (scattering, admittance or impedance).

2.1.1 Parameter-Dependent Basis Functions

The variations induced by the external parameters $\vartheta \in \Theta$ are embedded in the model structure (2.6) through the parameter-dependent basis functions $\xi_\ell(\vartheta)$. These basis functions must be selected carefully because upon this choice depends on the fitting accuracy. The literature offers several sets of functions, which are characterized by their own numerical properties.
In the following we will consider only one external parameter ($\rho = 1$).
One important point for our further observations is the (commonly used) procedure of improving the numerical conditioning of fitting algorithms by the normalization of the polynomials argument within $[-1,\ 1]$. In particular, we compute the normalized parameter value $\tilde{\vartheta}$ as:

$$\tilde{\vartheta} = -1 + 2 \cdot \frac{\vartheta - \vartheta_{min}}{\vartheta_{max} - \vartheta_{min}} \,. \tag{2.9}$$

The problem conditioning will direct affect the parameter-dependent basis choice.
We now provide several examples of the available choices for the parameter-dependent basis.

Monomials

The simplest polynomial function that could be used to capture the parameter evolution is defined as the standard monomials basis functions [40]

$$\xi_\ell(\vartheta) = \vartheta^\ell, \tag{2.10}$$

where $\ell = 0, ..., \bar{\ell}$ (as defined in (2.6)) and $\bar{\ell}$ is the basis order.
We provide a numerical example, realizing a third-order basis as showed in Fig.2.1. This represents the most intuitive case for parameter-dependent basis definition, but this sort of basis function usually leads to the construction of an *ill-conditioned* fitting problem.

Chebychev Polynomials

We introduce here a new set of basis functions, namely the orthogonal polynomials [2], [28]. From [8], we know that any orthogonal polynomial can be expressed with the recurrence formula that reads:

$$\xi_{\ell+1}(\vartheta) = (\alpha_\ell \vartheta + \beta_\ell)\xi_\ell(\vartheta) + \delta_{\ell-1}\xi_{\ell-1}(\vartheta). \tag{2.11}$$

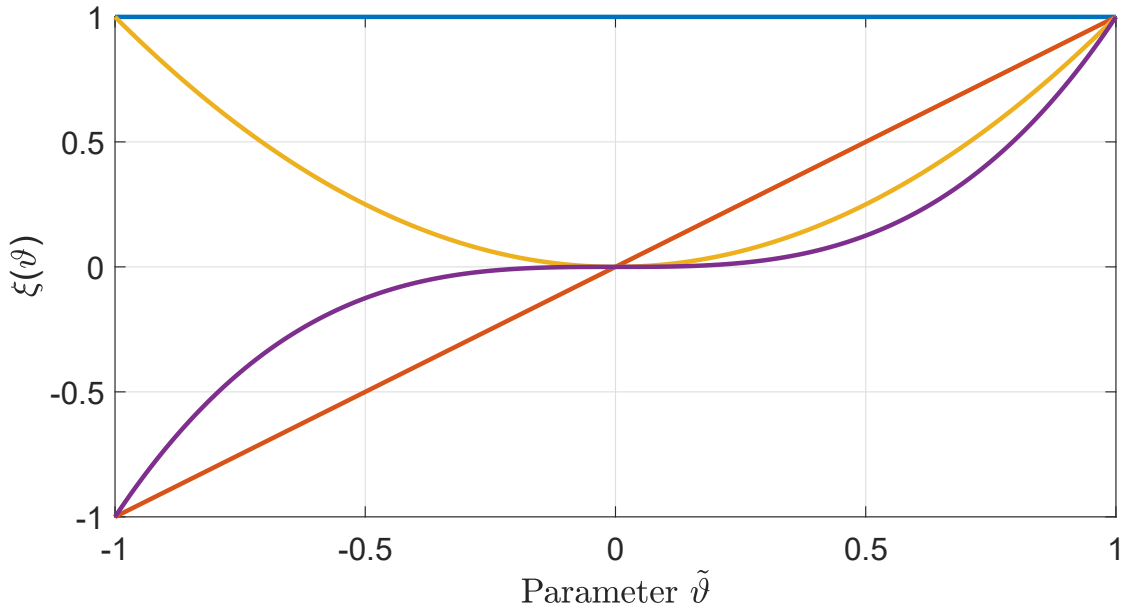


Figure 2.1: Monomials parameter-dependent basis evolution for $\ell = 0,1,2,3$.

In the following we will extensively use Chebychev polynomials, a special class of orthogonal polynomials, for which the expansion coefficients α, β, δ are equal to:

$$\alpha_0 = 1\,, \qquad \beta_0 = 0\,, \qquad \delta_0 = 0 \qquad \ell = 1\,, \tag{2.12}$$

$$\alpha_\ell = 2\,, \qquad \beta_\ell = 0\,, \qquad \delta_\ell = -1 \qquad \forall \ell \geq 1\,. \tag{2.13}$$

It is well known that the basis functions defined as before present very favourable numerical properties, which lead to a well-conditioned (and easy manageable) fitting problem. In this case, the regressor matrix created using such basis shows a reasonable condition number.
We denote the Chebychev polynomials of the first kind basis functions $\xi_\ell(\vartheta) = T_\ell(\vartheta)$ (see [5] and [13]) as:

$$T_\ell(\vartheta) = \cos[\ell \cos^{-1}(\vartheta)]\,, \quad \vartheta \in [-1,\, 1]\,, \quad \ell = 0, ..., \bar{\ell}; \tag{2.14}$$

which is equivalent to the standard expression

$$T_\ell(\cos t) = \cos(\ell t)\,, \quad t \in [0,\, 2\pi]\,, \quad \ell = 0, ..., \bar{\ell}. \tag{2.15}$$

An example of the fourth order Chebychev polynomials (first kind) is reported in Fig. 2.2.

Fourier Series

In order to guarantee a parameterization from a smooth function, when ϑ implies periodic variations, with $\vartheta \in [0,\, 2\pi]$ (e.g. the external parameter is an angle), as discussed in [18], we can define a parameter-dependent basis function as the standard Fourier basis in the

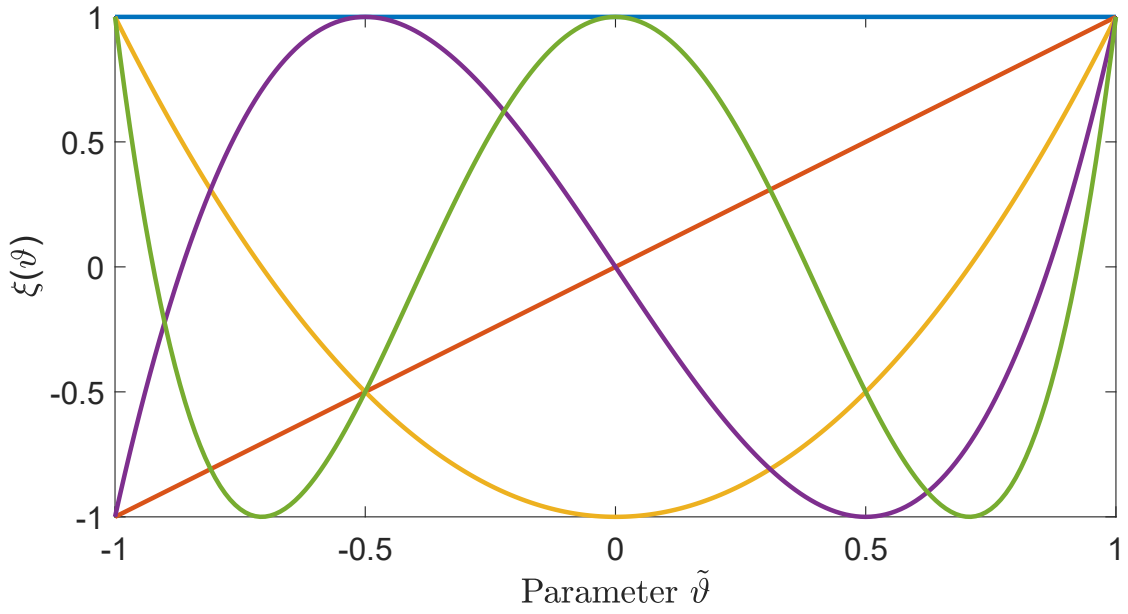


Figure 2.2: Chebychev parameter-dependent basis evolution for $\ell = 0,1,2,3$

trigonometric form

$$\xi_\ell(\vartheta) = \begin{cases} 1, & \ell = 0 \\ \cos(\lceil \ell/2 \rceil \vartheta), & \ell = 1,3,5,\ldots \\ \sin(\lceil \ell/2 \rceil \vartheta), & \ell = 2,4,6,\ldots \end{cases} \tag{2.16}$$

where the argument of $\lceil \cdot \rceil$ is rounded to the nearest larger integer. Figure 2.3 provides a numerical example for the first five terms of the Fourier basis ($\ell = 0,...,4$).

2.1.2 State Space and Descriptor Forms

We now present the state-space and descriptor realizations of a parameter-dependent LTI system, starting from the pole-residue form of the model $\mathbf{H}(s;\vartheta)$. As in the univariate case, also for a multivariate model this representation is appropriate to describe the properties of the model in algebraic form.

State Space Realizations

Following the procedure reported in Section 1.4.2, we can realize a parameter-dependent macromodel equivalent state-space description. Recalling the pole-residue model form of (1.35) and embedding the parameter dependency ϑ, the extension is straightforward.
In fact, considering the model of (2.7), with $\varphi_n(s)$ defined as the partial-fraction basis

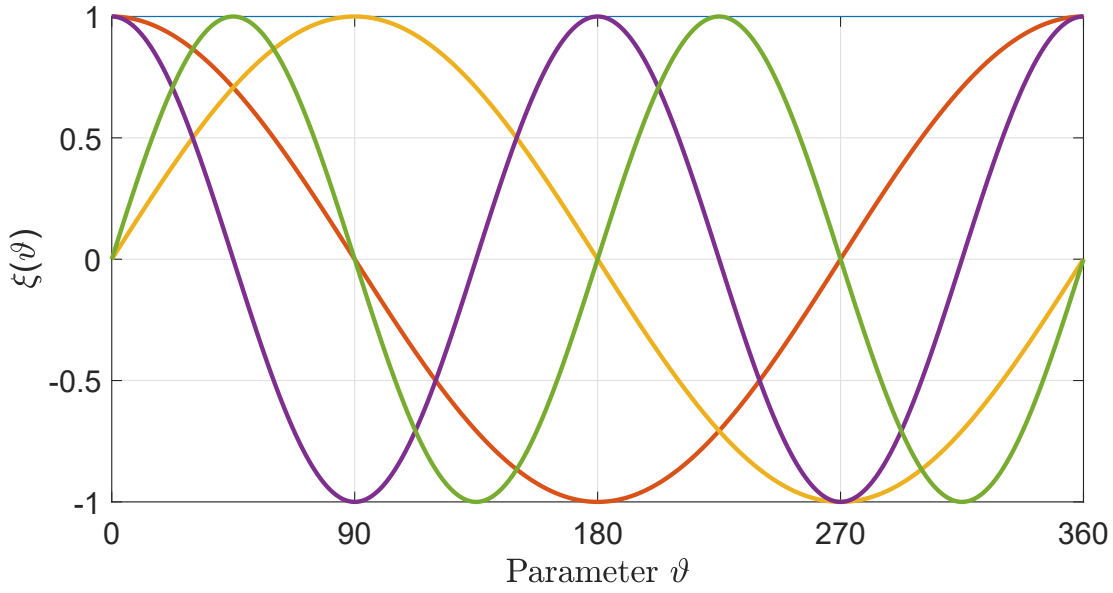


Figure 2.3: First four terms (ℓ = 0,1,2,3) of the Fourier parameter-dependent basis, through the parameter range $\vartheta \in [0°,360°]$. The polynomials arguments is normalized within $[-1,1]$ using the variable range.

with a prescribed set of real and complex poles q_n (see 1.2.1), we can write

$$\mathbf{N}(s,\vartheta) = \mathbf{R}_0(\vartheta) + \sum_{n=1}^{\bar{n}} \frac{\mathbf{R}_n(\vartheta)}{s-q_n} \tag{2.17}$$

$$\mathsf{D}(s,\vartheta) = r_0(\vartheta) + \sum_{n=1}^{\bar{n}} \frac{r_n(\vartheta)}{s-q_n}, \tag{2.18}$$

which allows us to construct the two separate state-space realizations for $\mathbf{N}(s,\vartheta)$ and $\mathsf{D}(s,\vartheta)$ as

$$\mathbf{N}(s,\vartheta) \leftrightarrow \left(\begin{array}{c|c} \mathbf{A}_0 & \mathbf{B}_0 \\ \hline \mathbf{C}_1(\vartheta) & \mathbf{D}_1(\vartheta) \end{array}\right) \tag{2.19}$$

$$\mathsf{D}(s,\vartheta)\mathbb{I}_P \leftrightarrow \left(\begin{array}{c|c} \mathbf{A}_0 & \mathbf{B}_0 \\ \hline \mathbf{C}_2(\vartheta) & \mathbf{D}_2(\vartheta) \end{array}\right), \tag{2.20}$$

where

$$\begin{aligned} \mathbf{A}_0 &= \text{blkdiag}\{\mathbf{A}_{0r}, \mathbf{A}_{0c}\} \\ \mathbf{B}_0^{\mathsf{T}} &= \left[\mathbf{B}_{0r}^{\mathsf{T}}, \mathbf{B}_{0c}^{\mathsf{T}}\right] \\ \mathbf{C}_1(\vartheta) &= \left[\mathbf{R}_1(\vartheta) \quad \cdots \quad \mathbf{R}_{\bar{n}}(\vartheta)\right] \\ \mathbf{C}_2(\vartheta) &= \left[\mathbb{I}_P\, r_1(\vartheta) \quad \cdots \quad \mathbb{I}_P\, r_{\bar{n}}(\vartheta)\right] \\ \mathbf{D}_1(\vartheta) &= \mathbf{R}_0(\vartheta) \end{aligned} \tag{2.21}$$

$$\mathbf{D}_2(\vartheta) = \mathbb{I}_P\, r_0(\vartheta). \tag{2.22}$$

with

$$\begin{aligned}
\mathbf{A}_{0r} =& \text{blkdiag}\{q_n \mathbb{I}_P\}_{n=1}^{\bar{n}_r} \\
\mathbf{A}_{0c} =& \text{blkdiag}\left\{ \begin{bmatrix} p'_n \mathbb{I}_P & p''_n \mathbb{I}_P \\ -p''_n \mathbb{I}_P & p'_n \mathbb{I}_P \end{bmatrix} \right\}_{n=1}^{\bar{n}_c} \\
\mathbf{B}_{0r} =& \left[1,\ldots,1\right]^{\mathsf{T}} \otimes \mathbb{I}_P \\
\mathbf{B}_{0c} =& \left[2,0,\ldots,2,0\right]^{\mathsf{T}} \otimes \mathbb{I}_P
\end{aligned} \tag{2.23}$$

Following the steps described in [36], we finally obtain the (compact) model state-space realization by the cascade of expression (2.19) as:

$$\mathbf{H}(s,\vartheta) = \mathbf{N}(s,\vartheta)(\mathsf{D}(s,\vartheta)^{-1}\mathbb{I}_P) \leftrightarrow \left(\begin{array}{c|c} \mathbf{A}_0 - \mathbf{B}_0\mathbf{D}_2^{-1}(\vartheta)\mathbf{C}_2(\vartheta) & \mathbf{B}_0\mathbf{D}_2^{-1}(\vartheta) \\ \hline \mathbf{C}_1(\vartheta) - \mathbf{D}_1(\vartheta)\mathbf{D}_2^{-1}(\vartheta)\mathbf{C}_2(\vartheta) & \mathbf{D}_1(\vartheta)\mathbf{D}_2^{-1}(\vartheta) \end{array} \right). \tag{2.24}$$

We recall [36] for more details.

Descriptor Forms

Recalling to the descriptor representation (1.57) of Section 1.4.2, we now define its parameter-dependent form [36] to (2.24).
The descriptor matrices, which now depend on the external parameter ϑ, are

$$\begin{aligned}
\mathbf{E} = \begin{pmatrix} \mathbb{I}_N & \mathbf{0}_{N,P} \\ \mathbf{0}_{P,N} & \mathbf{0}_{P,P} \end{pmatrix} \quad \mathbf{A}(\vartheta) = \begin{pmatrix} \mathbf{A}_0 & \mathbf{B}_0 \\ \mathbf{C}_2(\vartheta) & \mathbf{D}_2(\vartheta) \end{pmatrix} \\
\mathbf{C}(\vartheta) = \begin{pmatrix} \mathbf{C}_1(\vartheta) & \mathbf{D}_1(\vartheta) \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} \mathbf{0}_{N,P} \\ -\mathbb{I}_P \end{pmatrix}
\end{aligned} \tag{2.25}$$

with $\mathbf{0}_{J,K}$ null matrix of size $J \times K$. The other matrices of (1.58) denote the state-space realization of the model numerator $\mathbf{N}(s,\vartheta)$, described by the set $\{\mathbf{A}_0, \mathbf{B}_0, \mathbf{C}_1(\vartheta), \mathbf{D}_1(\vartheta)\}$, and the (extended) denominator $\mathsf{D}(s,\vartheta)\mathbb{I}_P$, described by $\{\mathbf{A}_0, \mathbf{B}_0, \mathbf{C}_2(\vartheta), \mathbf{D}_2(\vartheta)\}$, which are exactly the same elements of (2.19).
The model expression of (2.6) is equivalent to

$$\mathbf{H}(s;\vartheta) = \mathbf{C}(\vartheta)(s\mathbf{E} - \mathbf{A}(\vartheta))^{-1}\mathbf{B}\,, \tag{2.26}$$

as detailed in [36] and [16].

Chapter 3

Stability Enforcement

Our main objective is to realize a multivariate parametric macromodel $\mathbf{H}(s;\vartheta)$ which can be used for SI/PI circuit simulation: the identification process must be able to guarantee the model stability in the entire parameter domain Θ.
In this Chapter, we propose a strategy that is able to assure such condition by realizing a constrained fitting algorithm, which is based on the realization of a real positive model denominator. By starting from a finite set of samples of the true system response $\breve{\mathbf{H}}(\mathrm{j}\omega,\vartheta)$ we will able to generate robust and stable parameterized macromodels: these will be used in Chapter 5 and Chapter 6 to extract equivalent SPICE netlists for circuit simulation.
The procedure adopted here is not new [21], [9]. However, some fundamental improvements are presented, which guarantee uniform stability (different from previous approaches, which were not able to provide a stability certificate). These improvements are based on a final passivity enforcement, that is realized through a final loop on the denominator sub-model. Furthermore, a robust implementation was realized and it is here reported.
In the following, we provide numerical examples to illustrate the main characteristics of the procedure, by focusing on the stability enforcement. At the moment we will not stress the bias-dependent small signal systems, which will be the main subject of circuit simulations in later chapters.

3.1 Uniform Stability and PR Denominator

By using a model in the PSK-form (2.6), we adopt the (Parameterized) Sanathanan-Koerner (PSK) iterative procedure [21]. The related optimization problem can be summarized as

$$\min \left\| \frac{\mathsf{N}^{\mu}(\mathrm{j}2\pi f_k,\vartheta_m) - \mathsf{D}^{\mu}(\mathrm{j}2\pi f_k,\vartheta_m)\,\breve{\mathbf{H}}_{k,m}}{\mathsf{D}^{\mu-1}(\mathrm{j}2\pi f_k,\vartheta_m)} \right\| \tag{3.1}$$

which is a sequence of linear least-squares (LS) problems, where the denominator is initialized to $\mathsf{D}^0 = 1$. The minimization solution is achieved through estimations of the model coefficients $\mathbf{R}^{\mu}_{n,\ell}$ and $r^{\mu}_{n,\ell}$ at each iteration μ: the process terminates when the

model coefficients stabilize. If the solution converges, the final model provides a good approximation of the raw data response.

The above basic procedure does not ensure that the model poles have a strictly negative real part for possible values of the parameters ϑ. Moreover, from (2.6) we recall that a direct parameterization of the model poles $p_n(\vartheta)$ is not possible, due to a likely non-smooth behaviour of them when ϑ changes (see [18] for details). A direct stability condition can be stated as

$$\mathrm{Re}\left\{p_n(\vartheta)\right\} < 0 \quad \forall \vartheta \in \Theta\,, \quad n = 1, ..., \bar{n} \tag{3.2}$$

where $p_n(\vartheta)$ are the denominator zeros ($\mathsf{D}(p_n(\vartheta), \vartheta) = 0$).
Indeed, since the frequency-dependent basis poles q_n are actually only apparent singularities of the model (see Sec. 1.2.1), we can focus on the denominator sub-system and investigate only its properties (namely, its zeros). We refer to the denominator expansion series as

$$r_n(\vartheta) = \sum_{\ell=1}^{\bar{\ell}} r_{n,\ell}\, \xi_\ell(\vartheta) \tag{3.3}$$

where $r_{n,\ell}$ are the model coefficients.

We can now observe that

- assuming $\mathsf{D}(s, \vartheta)$ as a passive immittance function (Positive Real, PR), then its inverse $\mathsf{D}^{-1}(s, \vartheta)$ is also a passive immittance function;
- any (rational) PR function has stable real poles.

These concepts lead us to the conclusion that if we want to ensure the uniform model stability it is sufficient to define a PR denominator $\mathsf{D}(s, \vartheta)$ for any $\vartheta \in \Theta$: its zeros, which are the model poles $p_n(\vartheta)$, will be guaranteed stable independently by ϑ.

By recalling the definition of a Positive Real transfer matrix (Def. 1.3), we see for the case of a parameter-dependent scalar denominator (3.3) that:

1. $\mathsf{D}(s, \vartheta)$ is regular for $\mathrm{Re}\left\{s\right\} > 0$ since the poles q_n of the frequency-dependent basis function $\varphi_n(s)$ are stable by construction;
2. $\mathsf{D}^*(s, \vartheta) = \mathsf{D}(s^*, \vartheta)$ is guaranteed by construction.

Only the third condition is not granted so that a sufficient condition that guarantees a PR denominator is the following

$$\mathrm{Re}\left\{\mathsf{D}(\mathrm{j}\omega, \vartheta)\right\} > 0\,, \quad \forall \omega\,, \forall \vartheta \in \Theta\,. \tag{3.4}$$

A set of constraints must be realized to impose such condition at each iteration of the fitting algorithm (3.1).
By sampling (3.4) through an adaptive procedure, we can obtain a set of points to realize the linear inequalities constraints as

$$\mathrm{Re}\left\{\mathsf{D}(\mathrm{j}\omega_i, \vartheta_i)\right\} > 0\,, \tag{3.5}$$

which are reformulated during each iteration.

We now focus on the sampling procedure, which is necessary to proceed with the stability enforcement.

3.1.1 Sampling Process For Constraints Realization

In order to formulate the inequalities constraints as in (3.5), a set of points (ω_i, ϑ_i) must be created at each iteration μ of the optimization problem.
One of the most interesting improvements of this work, presented in [9], is provided by the automatic sampling procedure, based on two set of points:

1. a set of fixed points $\mathcal{F} = \{(\mathrm{j}\omega_k, \vartheta_m)\,, k = 1, ..., K\,, m = 1, ..., M\}$, which includes all the points from the raw data responses $\breve{\mathbf{H}}(\mathrm{j}\omega_k, \vartheta_m)$, both as frequencies or parameters values; we denote with n_f the dimension of this first set;

2. a set of adaptive-sampled points denotes as $\mathcal{W}^\mu = \{(\mathrm{j}\omega_j, \vartheta_j)\,, j = 1, ..., J\}$, which are provided searching the local minima in the denominator passivity violation regions of the (ω, ϑ) space; we identify with n_a^μ the overall number of components (at the μ-th iteration) of this set;

The combination of the above sets defines

$$\mathcal{Z}^\mu = \mathcal{F} \cup \mathcal{W}^\mu \tag{3.6}$$

which is the overall set of points at each μ iteration, of dimension $n_f + n_a^\nu$. We denote the elements of this data-set as

$$z_i^\mu = (\mathrm{j}\omega_i, \vartheta_i) \in \mathcal{Z}^\mu\,, \quad i = 1, ..., I \tag{3.7}$$

We now focus on the adaptive search process, previously introduced in [21]. In particular, we note that looking for the local minima point, the parameter ϑ span is restricted to the interval Θ, and the frequency domain is unlimited: the denominator may present a passivity violation outside the modelling bandwidth Ω. This corresponds to

$$\mathrm{Re}\left\{\mathsf{D}(\mathrm{j}\omega_j, \vartheta_j)\right\} < 0 \tag{3.8}$$

where $\vartheta_j \in \Theta$ and $\omega_j \notin \Omega$ identify a passivity violation point in the (ω, ϑ) space.
A Hamiltonian eigenvalue computation solves this problem since all frequencies ω_j where the real part of the denominator response crosses the zero baseline are purely imaginary eigenvalues $\mathrm{j}\omega_j$ of the Skew-Hamiltonian/Hamiltonian Pencil

$$\mathbf{M}(\vartheta) = \begin{pmatrix} \mathbf{A} & \mathbf{0} & \mathbf{B} \\ \mathbf{0} & -\mathbf{A}^\mathsf{T} & -\mathbf{C}^\mathsf{T}(\vartheta) \\ \mathbf{C}(\vartheta) & \mathbf{B}^\mathsf{T} & 2D(\vartheta) \end{pmatrix}, \qquad \mathbf{K} = \begin{pmatrix} \mathbb{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbb{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{3.9}$$

with $\big(\mathbf{A}, \mathbf{B}, \mathbf{C}(\vartheta), D(\vartheta)\big)$ state space representation matrices of the (immittance) denominator sub-model $\mathsf{D}(s, \vartheta)$ (2.20).

The automatic procedure that provides the second data-set $\mathcal{W}$ can be realized by following two strategies.
The first approach was built on the distance of the Hamiltonian eigenvalues set from the imaginary axis $\psi(\vartheta)$. Passivity violations points were founded by using this distance, which is larger than zero if the model is passive (otherwise is null), with a local-descendent

based search. This procedure is strongly affected by the initial coarse grid on Θ.
A second strategy solves this issues and provides a more efficient procedure to reach even better results in terms of local minima search. By using the parameter-dependent basis derivative (instead of $\psi(\vartheta)$) and the Hamiltonian eigenvalue computation, an automatic refinement of the sampling in the parameter space Θ is realized. Due to the intrinsic complexity of a multivariate case, the algorithm guarantees a successful procedure finding the passivity violations regions for two external parameters at most: this represents, at the present day, the best result available. We refer to A.Zanco thesis [44] and to [45] for details.

Nevertheless, the practical implementation of the proposed stability enforcement suffers from this limitation: the case with $\rho > 2$ is not supported yet.

By combining the above two data-sets and realizing $\mathcal{Z}$, we are now able to formulate a feasibility region for the constrained least-squares fitting problem which leads to a (guaranteed) stable and accurate model extraction.

3.2 Implementation of PR Strategy

We now focus on the practical implementation of the PR strategy to impose the uniform stability of a multivariate macromodel, by recalling to the T.Bradde thesis results [7]. Through his work, the Fast-PSK (or FPSK) scheme was implemented and the entire algorithm has achieved a significant speed-up. The stability enforcement implementation finds its place in this context, where a decoupling of the denominator coefficients in terms of numerator unknowns is realized. We provide a brief description of the FPSK scheme in the following (for details see [17]).

Starting from the PSK model structure (2.6), we can

- collect both the coefficients $r_{n,\ell}$ and $\mathbf{R}_{n,\ell}$ in a column vector $\boldsymbol{d} \in \mathbb{R}^{(\bar{n}+1)T}$ and $\boldsymbol{c}_{(i,j)} \in \mathbb{R}^{(\bar{n}+1)T}$, respectively, where we indicate with (i,j) the corresponding model response element;

- define a row vector function $g(s,\vartheta)$ collecting in its entries all combinations of frequency and parameter basis functions $\varphi_n(s)\,\xi_\ell(\vartheta)$ with a suitable ordering. We have that $g(s,\vartheta) \in \mathbb{C}^{1\times(\bar{n}+1)T}$;

- realize a matrix $\mathbf{\Phi} \in \mathbb{C}^{KM\times(\bar{n}+1)\bar{\ell}}$, where each row is the function $g(s,\vartheta)$ evaluated at each frequency and parameter values (s_k,ϑ_m), available from the raw data-set;

- define the matrix $\mathbf{W}^{\mu-1} = \text{blkdiag}\{\text{D}^{\mu-1}(\text{j}2\pi f_k,\vartheta_m)\}$

From the above steps, we can write the PSK-iteration minimization problem (3.1) in the (compact) least-squares matrix formulation as

$$\mathbf{\Psi}\boldsymbol{x} \approx \mathbf{0}\,, \tag{3.10}$$

where

$$\boldsymbol{\Psi} = \begin{pmatrix} \boldsymbol{\Gamma} & \mathbf{0} & \dots & \mathbf{0} & \boldsymbol{\Xi}_{(1,1)} \\ \mathbf{0} & \boldsymbol{\Gamma} & \dots & \mathbf{0} & \boldsymbol{\Xi}_{(2,1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \boldsymbol{\Gamma} & \boldsymbol{\Xi}_{(P,P)} \end{pmatrix}, \quad \boldsymbol{x} = \begin{pmatrix} \boldsymbol{c}_{(1,1)} \\ \boldsymbol{c}_{(2,1)} \\ \vdots \\ \boldsymbol{c}_{(P,P)} \\ \boldsymbol{d} \end{pmatrix}, \tag{3.11}$$

with $\boldsymbol{\Gamma} = \mathbf{W}^{\mu-1}\boldsymbol{\Phi}$ and $\boldsymbol{\Xi}_{(i,j)} = -\mathbf{W}^{\mu-1}\breve{\mathbf{H}}_{(i,j)}\boldsymbol{\Phi}$.
A QR-based application to the above least-squares system enables an efficient decoupling scheme as follows

$$(\boldsymbol{\Gamma}, \boldsymbol{\Xi}_\nu) = \mathbf{Q}_\nu \mathbf{R}_\nu = \mathbf{Q}_\nu \begin{pmatrix} \mathbf{R}_\nu^{11} & \mathbf{R}_\nu^{12} \\ \mathbf{0} & \mathbf{R}_\nu^{22} \end{pmatrix} \tag{3.12}$$

where ν denotes any pair (i, j), with $\nu = 1, ..., P^2$. At this point, we solve the least-squares system that enables the computation of all the denominator coefficients, stored in the vector $\boldsymbol{d}$, as

$$\begin{pmatrix} \mathbf{R}_1^{2,2} \\ \mathbf{R}_2^{2,2} \\ \vdots \\ \mathbf{R}_{P^2}^{2,2} \end{pmatrix} \boldsymbol{d} \approx \mathbf{0}\,. \tag{3.13}$$

Further orthogonalizations on the diagonal blocks of $\boldsymbol{\Gamma}$ optimize the QR-factorization and enable to compute all blocks in (3.12) within a loop over ν (see [17] for details). Another least-squares system, with multiple right-hand sides, computes the numerator coefficients at the end of the PSK-iterations as

$$\boldsymbol{\Gamma}\mathbf{C} \approx \mathbf{B}\,, \tag{3.14}$$

where

$$\mathbf{C} = (\boldsymbol{c}_{(1,1)}, \cdots, \boldsymbol{c}_{(P,P)}) \tag{3.15}$$

$$\mathbf{B} = (-\boldsymbol{\Xi}_{(1,1)}\boldsymbol{d}, \cdots - \boldsymbol{\Xi}_{(P,P)}\boldsymbol{d})\,. \tag{3.16}$$

The identification process defined through the FPSK-iteration does not guarantee the uniform stability of the resulting model. We can now focus on the stability enforcement embedded in the identification algorithm.

The novel contribution of the PR strategy for the model stability enforcement arises during the computation of the denominator unknowns vector $\boldsymbol{d}$ (3.13), starting from the second FPSK-iteration. Indeed, the inequality constraints (3.5) must be added to the above least-squares problem (3.13): its solution guarantees that the denominator sub-model is positive-real by construction.
This corresponds to the following process

1. check the denominator passivity of the model obtained from the previous iteration, denoted as $\mathrm{D}^{\mu-1}(\mathrm{j}2\pi f_k, \vartheta_m)$;

2. extract the passivity violation set $\mathcal{W}^\mu$ of n_a^μ adaptive points $(\mathrm{j}\omega_j, \vartheta_j)$, ;

3. realize the complete data set $\mathcal{Z}^\mu$, by combining the fixed data-set $\mathcal{F}$ and the adaptive-sampled data-set $\mathcal{W}^\mu$;

4. realize (3.4) in a matrix form.

We now focus on the last of these steps.
By employing the elements of the extended data set $z_i^\mu = (s_i, \vartheta_i)$ (using the Laplace variable) and from (2.6), we define a row function $g_D(s, \vartheta)$ collecting terms $\varphi_n(s)\,\xi_{\ell_D}(\vartheta)$ such that

$$b_{i;n,\ell} = g_{D_{n,\ell}}(s_i, \vartheta_j) = \varphi_n(s_i)\,\xi_{\ell_D}(\vartheta_i) \tag{3.17}$$

which enables us to assemble the vector

$$\boldsymbol{b}_i^T = \left[b_{i;0,1}\,, \cdots, b_{i;n,\ell}\,, \cdots, b_{i;\bar{n},\bar{\ell}}\right]\ . \tag{3.18}$$

The above procedure is equivalent to evaluate the row function $g_D(s, \vartheta)$ in the points of the extended data set $\mathcal{Z}^\mu$, which must be recomputed at each FPSK-iteration μ. We can now realize the matrix that realizes (3.4) as

$$\mathbf{A}_{pr}^\mu = \mathrm{Re}\left\{\mathbf{B}^\mu\right\}\,, \quad \mathbf{B}^\mu = \begin{pmatrix} \boldsymbol{b}_1^T \\ \vdots \\ \boldsymbol{b}_i^T \\ \vdots \\ \boldsymbol{b}_I^T \end{pmatrix} \tag{3.19}$$

where $\mathbf{A}_{pr}^\mu \in \mathbb{R}^{(n_f+n_a)\times(\bar{n}+1)\bar{\ell}}$ is the constraint matrix of the optimization problem.
We can now apply the inequality constraints (3.5) to the least-squares problem (3.13) in a matrix form as

$$\mathbf{A}_{pr}^\mu\,\boldsymbol{d} > \boldsymbol{\alpha} \tag{3.20}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^{(n_a+n_f)\times 1}$.
We impose $\boldsymbol{\alpha} > \mathbf{0}$ in order both to provide some tolerance to the optimization problem and to avoid the critical case of a denominator with a real response close to the machine accuracy[1]. Even if we apply a conservative assumption by setting each element of the vector as $\alpha_i = 1$, this condition does not affect the general result of the procedure that still ensures a positive-real denominator in the parameter space Θ (3.5). Moreover, the cost function remains the same of the standard fitting algorithm (3.13) and still guarantees a good accuracy of the resulting model.

We can summarize the constrained minimization problem as

$$\min_{\boldsymbol{d}} \sum_{\nu=1}^{P^2} \left\|\mathbf{R}_\nu^{2,2}\,\boldsymbol{d}\right\|_2^2 \quad \text{s.t.} \quad \mathbf{A}_{pr}\,\boldsymbol{d} > \boldsymbol{\alpha}\,, \quad \nu = 1, ..., P^2\,, \tag{3.21}$$

[1] In MATLAB this value corresponds to $1e^{-15}$

which is a convex problem and its solution is straightforward.
After that, the denominator unknowns are computed through the constrained least-squares problem, the FPSK-iteration can advance normally.

A note of attention is placed here. We actually started our investigation by applying the linear inequality constraints (3.19) to the basic PSK-iteration scheme. The resulting models from this first implementation were still guaranteed stable but, unfortunately, the time required for the optimization was very high. For this reason, we abandoned the PSK implementation, by focusing on the improvement of the above strategy for the FPSK-scheme only. Nevertheless, we will present some numerical result of a denominator passivity enforcement in a PSK-interation, in order to compare the time required for the model generation with the two procedures.

3.2.1 Numerical Results

In this section, we apply our modelling scheme for selected examples in order to highlight the critical points of the above procedure. We are going to use the Test Cases that are reported in Appendix A: we refer to this part for details about the physical structures under investigation.

We now focus on the main aspects of the PR strategy, comparing the above procedure results with the standard fitting procedures, both for an FPSK and PSK scheme. Indeed, we are going to report the main drawbacks which lead to abandoning the enforcement on a PSK-iteration in the following.

Case 1
First, we investigate a 'simple' case, which is characterized by a resulting model (from a standard system identification) with a non-positive real denominator in the parameter space. Nevertheless, even if a fine sweep of the external parameter does not indicate any unstable poles, there is not a guarantee that the model will be stable irrespective of the value of $\vartheta \in \Theta$.
Left panels of figure 3.1 show the results from a standard FPSK model generation, without imposing the passivity constraints (3.5), which requires 1.5 seconds to extract the final result. The bottom panel confirms that there exist some points in the space (ω, ϑ) such that $\mathrm{Re}\{\mathrm{D}(\mathrm{j}\omega, \vartheta)\} < 0$. The same results are achieved with a standard PSK extraction that took 9.75 seconds to generate the overall model.
The maximum absolute and relative errors for all the ports and parameters are reported in Table 3.1. Right panels of Fig. 3.1 compares the above results with the model generated from a constrained FPSK-scheme: the overall accuracy is maintained to all model ports. Moreover, the denominator model does not present any passivity violations in the parameter space Θ: bottom right panel of Fig. 3.1 confirms this assumption through a fine sweep on the parameter domain. The overall model extraction requires only 22.9 seconds.
This runtime is significantly smaller with respect to the time required by embedding the same procedure in a basic PSK-scheme, which took 601 seconds. This gap cannot be attributed to the two different approaches used to the passivity violations regions search: by using a derivative-based approach or a Hamiltonian distance, the same effects are

achieved both in terms of local minima results and elapsed time.
Moreover, the last generation produces a final model with a denominator that is still not PR for all the space (ω, ϑ).
We are going to further investigate this last point with the next example.

	Validation		Fitting	
	abs	rel	abs	rel
FPSK	2.98e-03	4.38e-03	1.86e-03	2.73e-03
FPSK with PR	3.05e-03	4.47e-03	1.80e-03	2.64e-03
PSK with PR	2.98e-03	4.38e-03	1.85e-03	2.72e-03

Table 3.1: Maximum absolute and relative errors on validation and fitting points, for all the parameters and ports, for **Case 1** of Appendix A.

Case 2
We now analyse a system which results in an unstable model from the standard fitting procedures, both FPSK (0.80 seconds) and PSK (0.74 seconds), and which shows a weakness of the proposed model generation.
In order to obtain a final positive-real denominator, we have applied first the constrained PSK-scheme and in a second step the constrained FPSK-scheme: they required 16.4 and 17.4 seconds, respectively, to extract a final model that is not guaranteed stable. Indeed, Figure 3.2 compares the real part of the two denominators responses and reports the corresponding Hamiltonian spectral distance from the imaginary axis: a null value of the curve corresponds to a denominator passivity violations region (see Section 3.1.1).
Top panels of Fig. 3.2 present these results.
The differences between the constrained FPSK and PSK cannot be attributed to the distinct sampling procedure used for the local minima search by building the constraints (3.20), which for this case provide the same passivity violations points. Nevertheless, this fact is out of the scope of this example, which is used to prove that both the procedures are not sufficient to provide a stable guaranteed result for this particular case.

Otherwise, the effectiveness of the proposed model generation is demonstrated by a variation of the same test case. Indeed, by imposing as an option the minimization of the relative error (for details see [7]), the model results guaranteed stable through a positive real denominator and the fitting accuracy is not compromised. Figure 3.3 shows the validation of the two models obtained through a standard FPSK (left) and the same scheme with the embedded stability enforcement (right): their extraction require 0.80 and 14.4 seconds, respectively. The absolute and relative errors for all the ports and parameters are reported in Table 3.2.

From these examples, it is clear that the uniform stability is still not guaranteed in general, and some more advanced enforcement is required. This is the subject of next section.

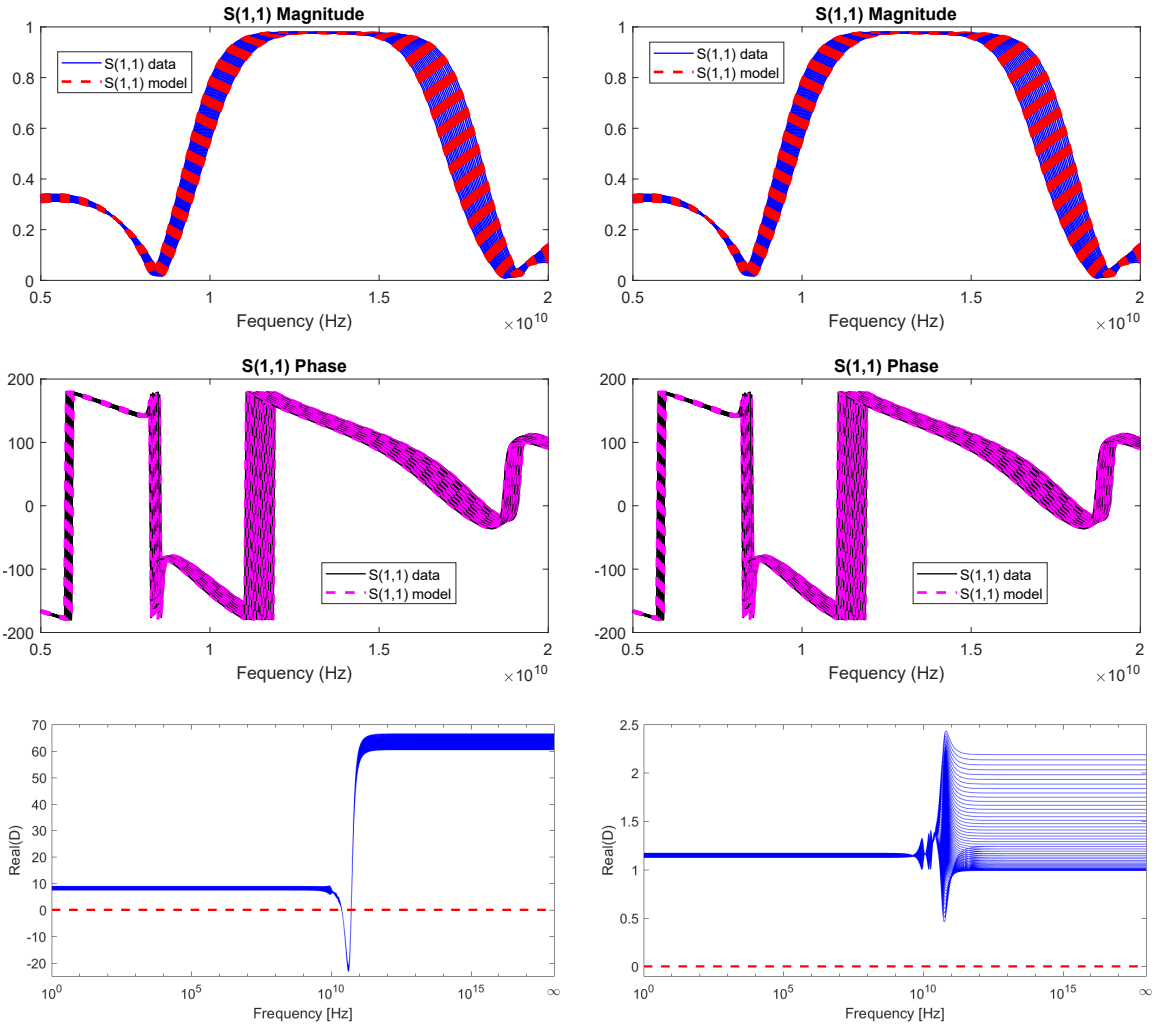


Figure 3.1: Validation of selected parameterized macromodel for **Case 1**.
Left Panels, standard FPSK: (unstable) model responses and (bottom) real part of the model denominator, computed over a fine sweep of the parameter value.
Right Panels, FPSK with PR enforcement: (stable) model responses after stability enforcement and (bottom) real part of the model denominator. Since $\mathrm{Re}\{\mathrm{D}(\mathrm{j}\omega, \vartheta)\} > 0$, it is guaranteed that the model poles are stable over $\vartheta \in \Theta$.

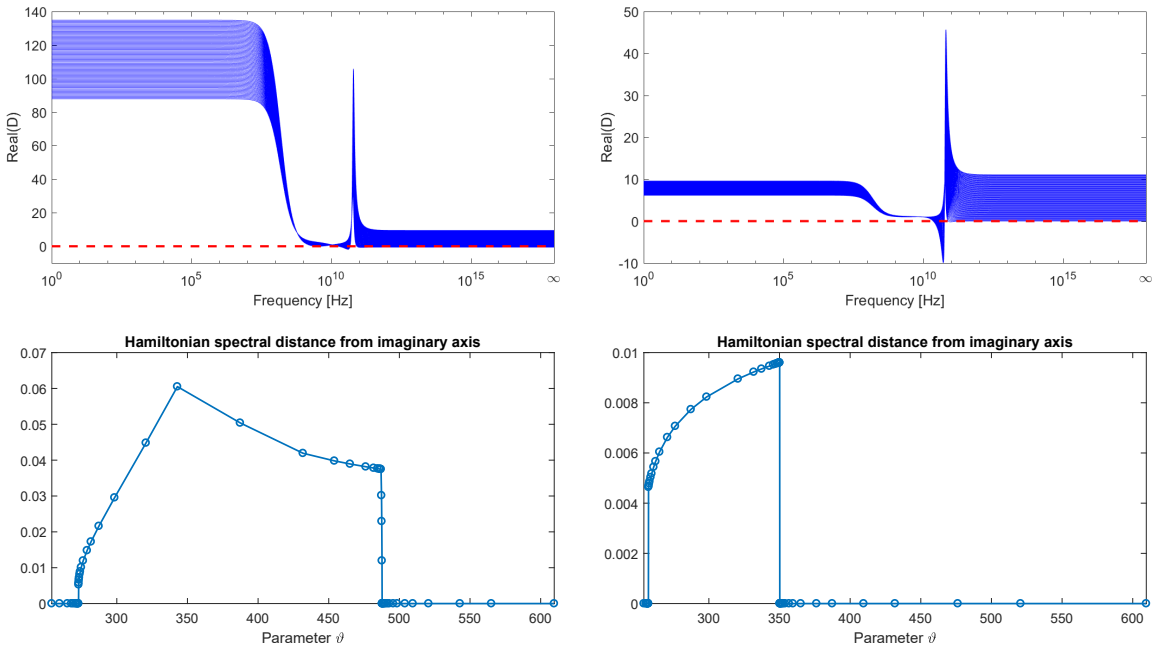


Figure 3.2: **Case 2** of Appendix A results. Left Panels, PSK with PR enforcement: (top) real part of the final model denominator, computed over a fine sweep of the parameter value; (bottom) Hamiltonian spectral distance from the imaginary axis, which corresponds to a passivity violation region when the curve assumes a null value.
Right Panels, FPSK with PR enforcement: (top) and (bottom) figure as above.

	Validation		Fitting	
	abs	rel	abs	rel
FPSK	8.25e-03	1.25e-02	8.25e-03	1.31e-02
FPSK with PR	7.63e-03	1.16e-02	8.35e-03	1.17e-02

Table 3.2: Maximum absolute and relative errors on validation and fitting points, computed over all the parameter values and port responses, for **Case 2** macromodel.

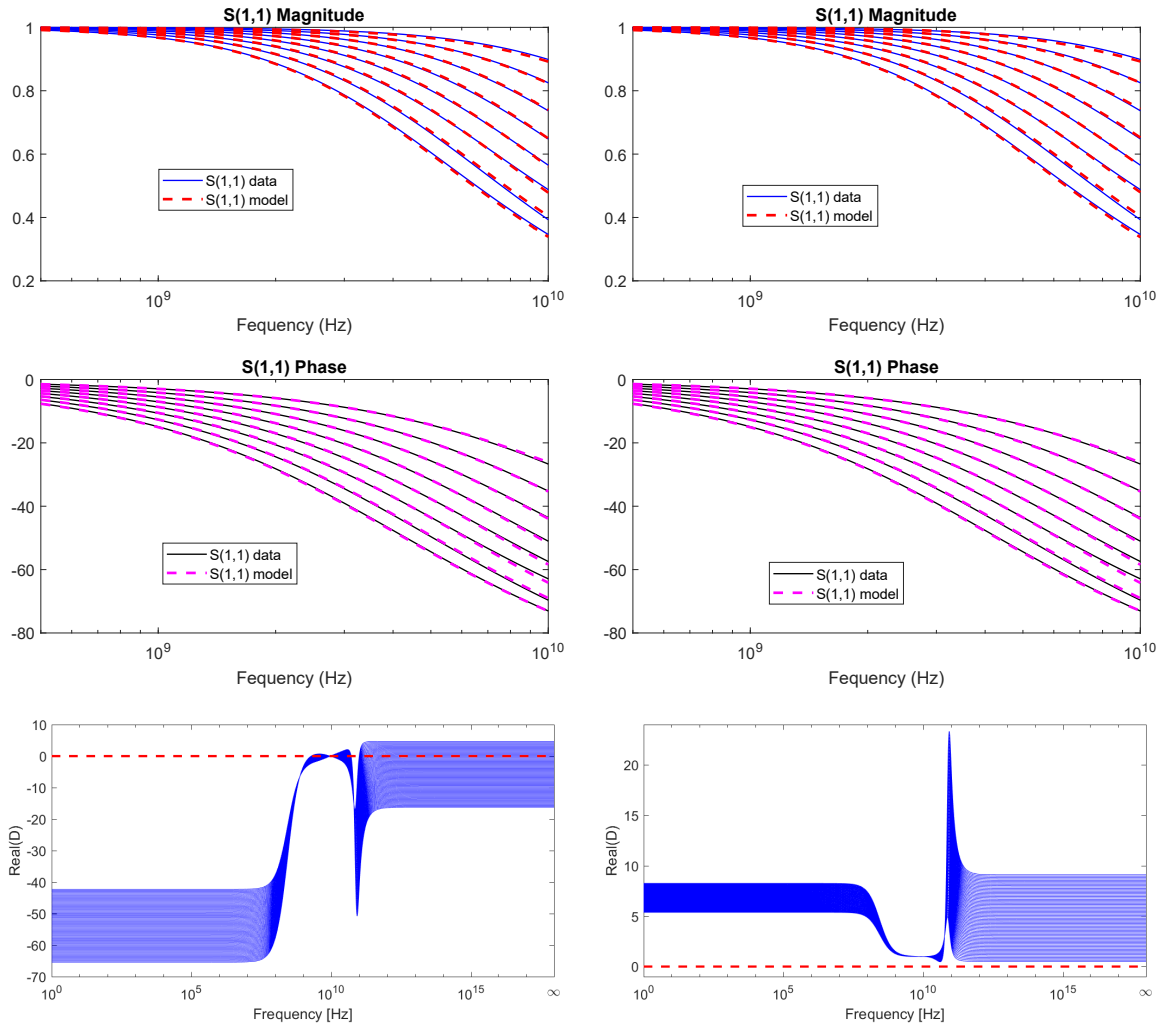


Figure 3.3: Validation of selected parameterized macromodel for **Case 2**, imposing the minimization of the relative error.
Left Panels, standard FPSK: (unstable) model responses and (bottom) real part of the model denominator.
Right Panels, FPSK with PR enforcement: (stable) model responses after stability enforcement and (bottom) real part of the model denominator. Since $\mathrm{Re}\{\mathrm{D}(\mathrm{j}\omega, \vartheta)\} > 0$, it is guaranteed that the model poles are stable over $\vartheta \in \Theta$.

3.3 Final Stability Enforcement on Denominator

By embedding positive real denominator constraints at each iteration of the identification process is actually not a sufficient condition to guarantee the stability of the resulting model. Indeed, the last PSK-iteration may realize a model which could actually be not PR: some passivity violation regions of the denominator may be still present. To eliminate this possibility, a final bivariate passivity enforcement based on first order-perturbation is realized on the denominator sub-model after the last fitting iteration. We now focus on this procedure [16], providing some additional details.

A perturbed denominator based on (2.6) is defined as

$$\widehat{\mathsf{D}}(s,\vartheta) = \mathsf{D}(s,\vartheta) + \Delta\mathsf{D}(s,\vartheta) \tag{3.22}$$

with

$$\Delta\mathsf{D}(s,\vartheta) = \sum_{n=0}^{\bar{n}} \sum_{\ell=1}^{\bar{\ell}} \Delta r_{n,\ell}\, \xi_\ell(\vartheta)\, \varphi_n(s) \tag{3.23}$$

We compute the coefficients perturbations $\Delta r_{n,\ell}$ such that the response of the perturbed denominator will be PR $\forall\vartheta$, satisfying the condition (3.4). These unknowns are located in a vector of decision variables denoted as

$$\boldsymbol{x} = \left[\Delta r_{0,0}\,, \cdots, \Delta r_{n,\ell}\,, \cdots, \Delta r_{\bar{n},\bar{\ell}}\right]^T \quad \text{with} \quad \boldsymbol{x} \in \mathbb{R}^{(\bar{n}+1)\bar{\ell}}\,. \tag{3.24}$$

We formulate the algebraic passivity constraints similarly to what we did in Section 3.2. In particular, we create the constraint matrix $\mathbf{A}_{pr}^{\mu}$ at each iteration of the passivity enforcement μ, by evaluating the function $g_D(s,\vartheta)$ which collects $\varphi_n(s)\,\xi_\ell(\vartheta)$ in the points $(\mathrm{j}\omega_i, \vartheta_i)$ of the complete set $\mathcal{Z}$, which is obtained combining the passivity violation check results and the fixed points from the raw data-set.

Moreover, to formulate the inequality constraints we have to consider that our goal is to realize a positive real final model $\widehat{\mathrm{D}}(s,\vartheta)$. This condition read as

$$\mathbf{A}_{pr}^{\mu}\widehat{\boldsymbol{x}} > \boldsymbol{\alpha} \tag{3.25}$$

where $\boldsymbol{\alpha}$ is defined as in (3.20) and $\widehat{\boldsymbol{x}}$ is the vector that stores the perturbed model coefficients. From (3.13) and (3.24), we can denote it as

$$\widehat{\boldsymbol{x}} = \boldsymbol{d} + \boldsymbol{x}\,. \tag{3.26}$$

By combining (3.25), and (3.26) we obtain the following expression

$$\mathbf{A}_{pr}^{\mu}\,\boldsymbol{d} + \mathbf{A}_{pr}^{\mu}\,\boldsymbol{x} > \boldsymbol{\alpha}\,. \tag{3.27}$$

From (3.19) we can see that $\mathbf{A}_{pr}^{\mu}\,\boldsymbol{d} = \mathrm{Re}\{\mathrm{D}(\mathrm{j}\omega_i, \vartheta_i)\}$, and we can define the resulting vector as

$$\boldsymbol{w}^{\mu} = \mathrm{Re}\left\{\mathsf{D}^{\mu}(\mathrm{j}\omega_i, \vartheta_i)\right\} \quad \text{with} \quad i = 1, ..., I\,, \tag{3.28}$$

The final inequality constraint can be cast in an algebraic form as

$$\mathbf{A}_{pr}^{\mu}\,\boldsymbol{x} > \tilde{\boldsymbol{\alpha}}^{\mu} \tag{3.29}$$

with

$$\tilde{\boldsymbol{\alpha}}^{\mu} = \boldsymbol{\alpha}^{\mu} - \boldsymbol{w}^{\mu} \tag{3.30}$$

where $\tilde{\boldsymbol{\alpha}}^{\mu} \in \mathbb{R}^{n_f+n_a} \times 1$ realizes an adaptive threshold that considers the denominator passivity violations regions at each enforcement iteration.

A proper cost function must be realized to guarantee that accuracy is preserved, by recalling to the raw data set $\breve{\mathbf{H}}_{k,m}$, as

$$\mathcal{E}^2 = \sum_{k=1}^{K}\sum_{m=1}^{M} |\Delta\mathsf{D}(s_k, \vartheta_m)|^2 \tag{3.31}$$

By referring the function $g(s, \vartheta)$, we can now define the single element

$$b_{k,m;n,\ell} = g_{n,\ell}(s_k, \vartheta_m) = \varphi_n(s_k)\xi_\ell(\vartheta_m) \tag{3.32}$$

of the vector denoted as

$$\boldsymbol{b}_{k,m}^{T} = \left[b_{k,m;0,1}\,, \cdots, b_{k,m;n,\ell}\,, \cdots, b_{k,m;\bar{n},\bar{\ell}}\right]\,. \tag{3.33}$$

This enables us to cast the cost function as

$$\mathcal{E}^2 = \left\|\tilde{\mathbf{F}}\,\boldsymbol{x}\right\|_2^2 \tag{3.34}$$

where

$$\tilde{\mathbf{F}} = \begin{Bmatrix} \mathrm{Re}\,\{\mathbf{F}\} \\ \mathrm{Im}\,\{\mathbf{F}\} \end{Bmatrix}\,, \qquad \mathbf{F} = \begin{Bmatrix} \boldsymbol{b}_{1,1}^{T} \\ \vdots \\ \boldsymbol{b}_{m,k}^{T} \\ \vdots \\ \boldsymbol{b}_{M,K}^{T} \end{Bmatrix} \tag{3.35}$$

Since $\tilde{\mathbf{F}} \in \mathbb{R}^{2KM\times(\bar{n}+1)\bar{\ell}}$ collects a number of rows equal to the available frequencies and parameters samples and usually $2KM \gg (\bar{n}+1)\bar{\ell}$, a QR-factorization is suggested to both changes the unknown variable and to reduces the optimization problem dimension. We can see that, using $\tilde{\mathbf{F}} = \mathbf{Q}\boldsymbol{\Psi}$ with $\mathbf{Q}^T\mathbf{Q} = \mathbb{I}$, the cost function can be written as

$$\mathcal{E}^2 = \|\boldsymbol{\Psi}\boldsymbol{x}\|_2^2 = \|\boldsymbol{y}\|_2^2\,. \tag{3.36}$$

This procedure implies to change accordingly the constraint definition, by applying the following transformation to the condition matrix

$$\tilde{\mathbf{A}}_{pr}^{\mu} = \mathbf{A}_{pr}^{\mu}\boldsymbol{\Psi}^{-1} \tag{3.37}$$

so that the resulting constrained minimization problem at each μ iteration reads

$$\min \|\boldsymbol{y}\|_2^2 \quad \text{s.t.} \quad \tilde{\mathbf{A}}_{pr}\,\boldsymbol{y} > \tilde{\boldsymbol{\alpha}} \tag{3.38}$$

with

$$\boldsymbol{x} = \boldsymbol{\Psi}^{-1}\boldsymbol{y}\,. \tag{3.39}$$

This represents a convex problem with a direct solution. Nevertheless, the first order approximation may require that this optimization has to be repeated until all the passivity violations are removed, and so until the denominator sub-model is guaranteed to be passive (and so Positive Real).
The Algorithm 3.1 shows the entire process in a pseudocode form: a maximum number of passivity enforcement iterations δ is imposed to extract a final model with a restricted number of perturbations. Indeed, with the continuation of the final enforcement on the denominator sub-model, the accuracy of the fitting process is reduced, as we are going to detail in the numerical examples.

Algorithm 3.1 Final Stability Enforcement (PR) Algorithm

Require: raw data $\breve{\mathbf{H}}(\mathrm{j}\omega, \vartheta)$
Require: frequency basis φ_n for $n = 1, \ldots, \bar{n}$;
Require: parameter basis ξ_ℓ for $\ell = 1, ..., \bar{\ell}$;
Require: non-positive real denominator $\mathsf{D}(s, \vartheta)$ coefficients $r_{n,\ell}$;
Require: maximum number of iterations δ
Set $\mu = 0$
Get current denominator passivity violations vector $\mathcal{W}^\mu$ as in Section 3.1.1
while $\mathcal{W}^\mu$ is not empty or $\mu < \delta$ **do**
    build complete data set $\mathcal{Z}^\mu = \mathcal{F} \cup \mathcal{W}^\mu$
    build constraint matrix A^μ_{pr} (3.20) for each element in $\mathcal{Z}^\mu$
    build vector $\boldsymbol{w}^\mu$ as $\mathrm{Re}\{\mathsf{D}(s, \vartheta)\}$ for each element in $\mathcal{Z}^\mu$
    build vector $\tilde{\boldsymbol{\alpha}}$ in (3.30)
    build matrix $\tilde{\mathbf{F}}$ in (3.35)
    build matrix $\boldsymbol{\Phi}$ in (3.36)
    build reduced constraint matrix $\tilde{\mathbf{A}}^\mu_{pr}$ in (3.37)
    solve convex optimization problem (3.38)
    evaluate denominator coefficients perturbation $\Delta r_{n,\ell}$ using (3.39)
    update $\mathsf{D}(s, \vartheta)$ coefficients as $r_{n,\ell} \leftarrow r_{n,\ell} + \Delta r_{n,\ell}$
    $\mu \leftarrow \mu + 1$
    get passivity violations $\mathcal{W}^\mu$ of the perturbed denominator model
end while
return PR denominator $\widehat{\mathsf{D}}(s, \vartheta)$ and number of iterations μ

After the denominator is guaranteed Positive Real, the numerator model coefficients are re-optimized, by considering the new (perturbed) denominator which is guaranteed passive. In this case, we adopt a standard procedure that computes, through a simple linear least-squares system, the numerator coefficients by 'removing' the denominator contribution from the equation. We can define

$$\underline{\boldsymbol{\Phi}}\mathbf{C} \approx \breve{\mathbf{H}} \tag{3.40}$$

where $\mathbf{C}$ is the numerator coefficients vector (3.14) and $\underline{\mathbf{\Phi}}$ is a matrix defined from (3.11) as

$$\underline{\mathbf{\Phi}} = \frac{\mathbf{\Phi}}{\mathrm{D}(s_k, \vartheta_m)} \quad k = 1, ..., K;\ m = 1, ..., M\,. \tag{3.41}$$

By evaluating the denominator model, the ordering of (s_k, ϑ_m) must be chosen to accord the one selected for $\mathbf{\Phi}$ and $\mathbf{C}$.
In order to ensure real-valued numerator coefficients, a standard re-arranging of the above least-squares matrices by splitting their real and complex contributions is required.

We recall that this strategy is an extension of a standard passivity enforcement procedure [16] and that its implementation to a multivariate case is straightforward.

The above procedure was not implemented also in a basic PSK-scheme due to the significant time required for the model extraction.

3.3.1 Numerical Results

In this section, we provide numerical results that demonstrate the effectiveness of the above final stability enforcement. We refer to Appendix A for the description of the structures under investigation.

Case 2
We now apply the final stability enforcement to the same test case of Sec. 3.2.1. In this example, we do not provide any comparison with the PSK-scheme, which was not implemented also with this final passivity enforcement.
An FPSK extraction provides a resulting model which is now guaranteed stable, as Figure 3.4 demonstrates, with an overall elapsed time for the model extraction of 18.1 seconds. Only two final passivity enforcement iterations are necessary, in this case, to obtain a still acceptable result, which actually shows a slightly degraded fitting accuracy with respect to the non-stable model. Nevertheless, this is the only approach that leads to a stable final model. The maximum absolute and relative errors are reported in Table 3.3.

	Validation		Fitting	
	abs	rel	abs	rel
FPSK	8.61e-03	1.31e-02	7.47e-03	1.75e-02
FPSK with PR	1.01e-02	1.54e-02	1.16e-02	1.77e-02
PSK with PR	1.01e-02	1.54e-02	1.16e-02	1.76e-02

Table 3.3: Maximum absolute and relative errors on validation and fitting points, computed over all the parameter values and port responses, for **Case 2** macromodel.

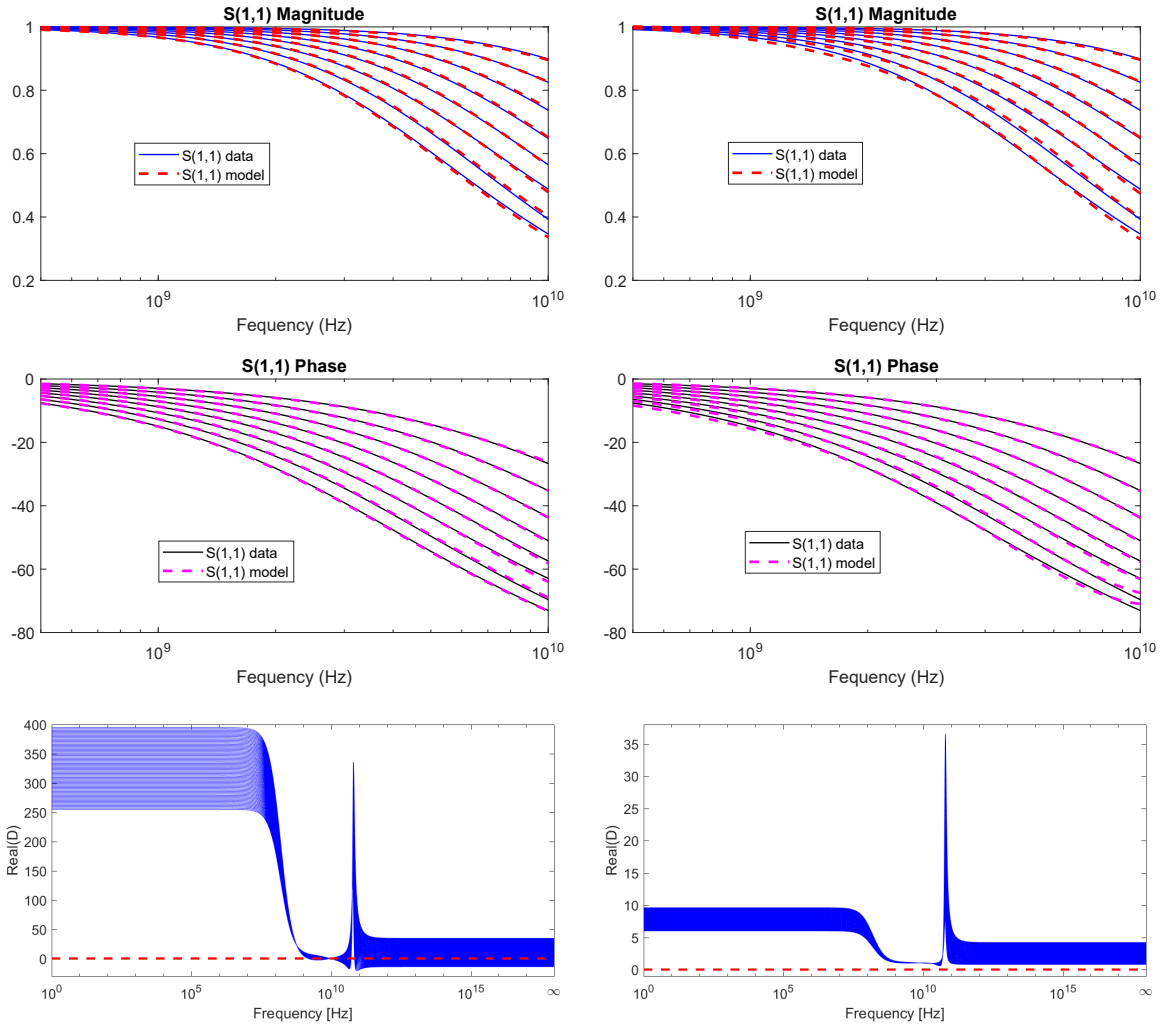


Figure 3.4: Validation of selected parameterized macromodel for **Case 2**.
Left Panels, standard FPSK: (unstable) model responses and (bottom) real part of the model denominator, computed over a fine sweep of the parameter value.
Right Panels, FPSK with PR enforcement: (stable) model responses after to two iterations of final stability enforcement and (bottom) real part of the model denominator. Since $\mathrm{Re}\{\mathrm{D}(\mathrm{j}\omega, \vartheta)\} > 0$, it is guaranteed that the model poles are stable over $\vartheta \in \Theta$.

Case 3

We now investigate a critical case that presents a non-convergent final passivity enforcement on the denominator. Nevertheless, we specify that this issue is not directly related to the proposed procedure, which effectiveness is demonstrated above all the other test cases used (and documented in Appendix A). Indeed, this specific case presents a recursive trend of the perturbed denominator during the final enforcement iterations: the algorithm as proposed above is not sufficient to guarantee a convergent solution in such conditions. Next section will present an optimization of the procedure that is focused on this subject.

Moreover, we document a system which requires all the available points in the parameter space during the fitting procedure, to obtain an acceptable result in terms of final model accuracy: this case indicates, in general, a critical condition for any fitting process.

The above statement holds also for the standard model generation strategy (FPSK), which requires 1.08 seconds to extract an unstable but sufficiently accurate model.

Nevertheless, after 20 iterations of the final stability enforcement, we are not still able to perturb sufficiently the denominator: the resulting model by interrupting the procedure is still not guaranteed stable. The entire extraction requires 61.86 seconds and produces the unstable model poles of Figure 3.5(top).

By proceeding with the final passivity enforcement on the denominator we discovered an increasing degradation of model accuracy: for this reason, we imposed a maximum number of iterations. Figure 3.5(bottom) compares the result from a standard model generation and from the constrained extraction stopped at the 20-th iteration. As it can be seen, the perturbed model is no more accurate for high frequencies.

From this example, it is clear that a convergent solution is not guaranteed in general. An advance improvement of the enforcement is the subject of next section.

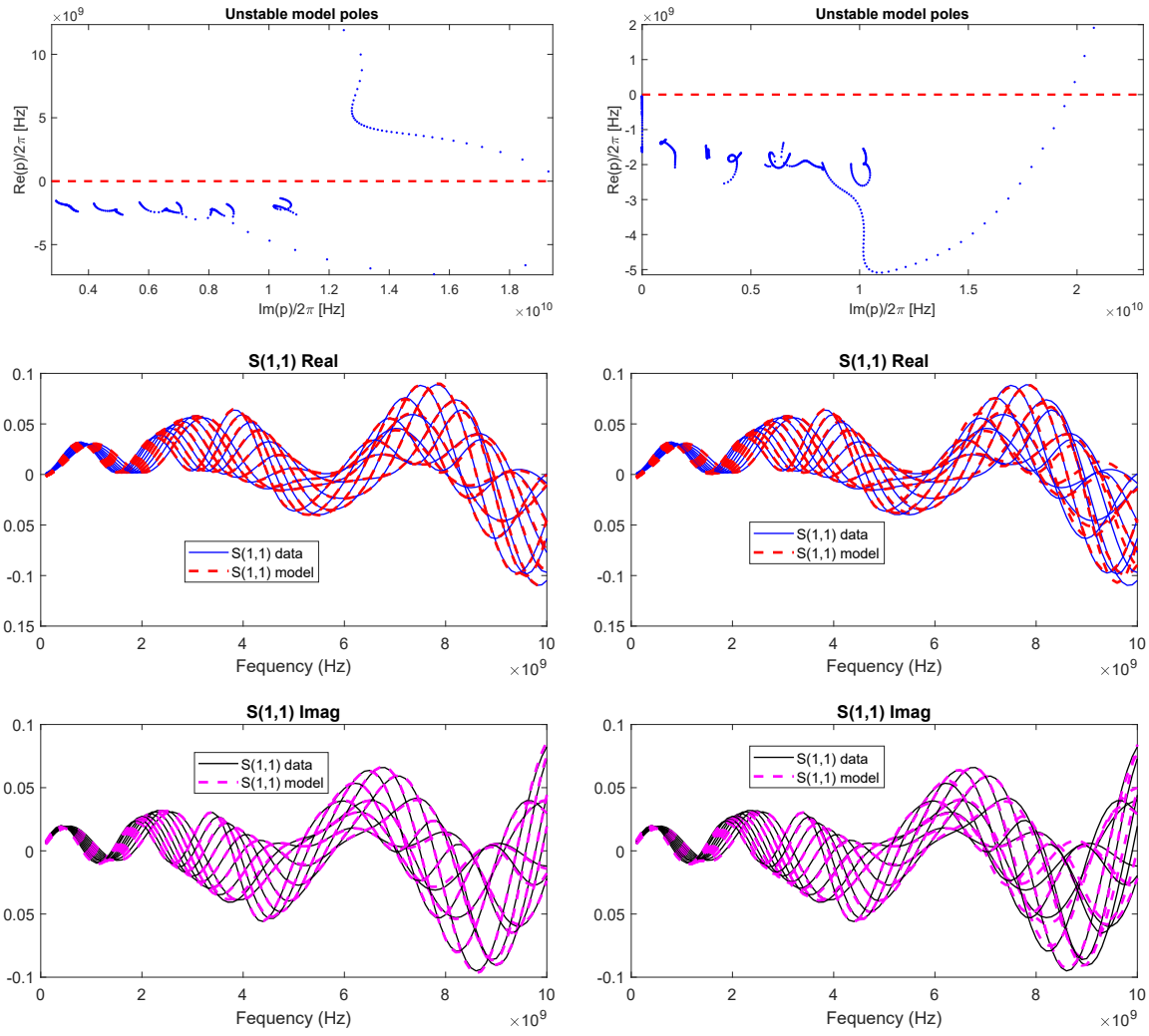


Figure 3.5: Validation of selected parameterized macromodel for **Case 3**. Left Panels, standard FPSK: (top) unstable parameterized model poles, computed over a fine sweep of the parameter value; (bottom) validation of the model responses. Right Panels, FPSK with denominator passivity enforcement, stopped at the 20-th iteration: (top) model poles still unstable; (bottom) validation of the model responses, which shows a degradation of model accuracy.

3.4 Robust Enforcement Implementation

The final passivity enforcement on the denominator sub-model enables to realize a positive-real system: nevertheless, sometimes many iterations are necessary in order to achieve this result and a convergent solution is not always guaranteed. To improve the algorithm we realized a 'robust' version of this enforcement, which is able to restrict the feasibility set of the optimization problem and provides an accurate final model.
Indeed, by using a 'predictive' approach for the passivity violation regions search, we are able to increase the number of the algorithm constraints during each iteration of the passivity enforcement. This procedure enables a better approximation of the feasibility region and ensures a convergent solution of the optimization problem.

The final passivity enforcement robust implementation is mainly composed of two nested loops.
An inner loop realizes the 'predictive' part of this algorithm through i iterations. Indeed, exactly the same passivity enforcement of Section 3.3 is performed, by changing the notations as follows

- the denominator model is defined as $\underline{\mathsf{D}}(s,\vartheta)$
- the perturbed coefficients of (3.24) are denoted as $\Delta\underline{r}_{n,\ell}^{(i)}$
- the data-sets of the i-th iteration are built as $\mathcal{Z}_i = \mathcal{F} \cup \mathcal{W}_i$.

The inner loop variables are initialized as

$$\underline{\mathsf{D}}^{(0)}(s,\vartheta) = \overline{\mathsf{D}}(s,\vartheta)\,, \quad \mathcal{W}_1 = \mathcal{W}^{\mu}\,, \tag{3.42}$$

where $\overline{\mathsf{D}}(s,\vartheta)$ and $\mathcal{W}^{\mu}$ are the denominator model and the sampled-data set from the external loop, respectively.

The innovative improvement provided by this internal procedure is that the passivity violations regions of the inner perturbed model are stacked, at the end of each i-th iteration, in an extended data-set denoted as $\mathcal{W}_{ext}$.
The 'predictive' loop ends when either the internal perturbed model $\underline{\mathsf{D}}(s,\vartheta)$ is passive or the maximum number of iterations γ is reached. We recall that the internal model $\underline{\mathsf{D}}(s,\vartheta)$ does not preserve any memory of the inner loop iterations and it is not affected by the extended data-set $\mathcal{W}_{ext}$.
At the end of the 'predictive' iterations, the new (extended) sampled data-set is passed to the external loop, which builds the constraints of (3.29) from

$$\mathcal{Z}^{\mu} = \mathcal{W}_{ext} \cup \mathcal{F}\,. \tag{3.43}$$

This procedure results in a 'restricted' feasibility set of the optimization problem (3.38) that leads to a completely different solution in terms of denominator coefficients perturbations, which are denoted now as $\Delta\overline{r}_{n,\ell}$. The resulting perturbed model $\overline{\mathsf{D}}(s,\vartheta)$ and the data-set $\mathcal{W}^{\mu}$, which is obtained from the local minima search at the end of each iteration of the external loop, are passed to the next μ iteration and, consequently, as inputs of the inner loop.

Algorithm 3.2 Robust Final Stability Enforcement (PR) Algorithm

Require: raw data $\breve{\mathbf{H}}(\mathrm{j}\omega, \vartheta)$
Require: frequency basis φ_n for $n = 1, \ldots, \bar{n}$;
Require: parameter basis ξ_ℓ for $\ell = 1, ..., \bar{\ell}$;
Require: non-positive real denominator $\mathsf{D}(s, \vartheta)$ coefficients $r_{n,\ell}$;
Require: maximum number of final iterations δ
Require: maximum number of predictive iterations γ
Set $\mu = 0$
Get current model passivity violations vector $\mathcal{W}$ as in Section 3.1.1
while $\mathcal{W}^\mu$ is not empty or $\mu < \delta$ **do**
    Set $\underline{\mathsf{D}}^{(0)}(s, \vartheta) = \overline{\mathsf{D}}(s, \vartheta)$ and $\mathcal{W}_1 = \mathcal{W}^\mu$
    Set $i = 1$ and $\mathcal{W}_{ext}$ as empty
    while $i \leq \gamma$ **do**
        Get perturbation coefficients $\Delta \underline{r}_{n,\ell}^{(i)}$ as in Algorithm 3.1, given $\mathcal{Z}_i = \mathcal{W}_i \cup \mathcal{F}$
        Update $\underline{\mathsf{D}}^{(i)}(s, \vartheta)$ coefficients as $\underline{r}_{n,\ell}^{(i+1)} \leftarrow \underline{r}_{n,\ell}^{(i)} + \Delta \underline{r}_{n,\ell}^{(i)}$ to get the perturbed intermediate model $\underline{\mathsf{D}}^{(i+1)}(s, \vartheta)$
        Get $\underline{\mathsf{D}}^{(i+1)}(s, \vartheta)$ passivity violations $\mathcal{W}_{i+1}$ as in Section 3.1.1
        if $\mathcal{W}_{i+1}$ is not empty **then**
            Stack passivity violations vector $\mathcal{W}_{i+1}$ in $\mathcal{W}_{ext} = [\mathcal{W}_{ext}^{\mathsf{T}},\ \mathcal{W}_{i+1}^{\mathsf{T}}]^{\mathsf{T}}$
        else
            break
        end if
        $i \leftarrow i + 1$
    end while
    Get perturbation coefficients $\Delta \overline{r}_{n,\ell}$ as in Algorithm 3.1 with the extended set of passivity violations $\mathcal{W}_{ext}$, such that $\mathcal{Z}^\mu = \mathcal{W}_{ext} \cup \mathcal{F}$
    Update $\overline{\mathsf{D}}(s, \vartheta)$ coefficients as $\overline{r}_{n,\ell} \leftarrow \overline{r}_{n,\ell} + \Delta \overline{r}_{n,\ell}$
    $\mu \leftarrow \mu + 1$
    Get passivity violations $\mathcal{W}^\mu$ of the perturbed model
end while
return PR denominator $\overline{\mathsf{D}}(s, \vartheta)$ and number of iterations μ

The procedure is repeated until the solution of the optimization problem converges or the maximum number of external iterations δ is reached.
Algorithm 3.2 shows the entire process in a pseudocode form.

The above procedure presents several drawbacks.
First, the positive real assumption is an only sufficient condition to the macromodel stability: even if the resulting model from a standard identification does not reveal any unstable pole, if its denominator is not PR the above enforced fitting procedure is required, also if (theoretically) this would be not necessary.
Unfortunately, at the present day, a uniform stability enforcement on the parameter space Θ which is able to overtake this issue does not exist and the only way to reveal the model unstable poles is to perform a brute force sampling through the parameter domain. The

PR strategy remains, for this reason, the only way to uniformly guarantee the stability of a parameterized macromodel.

Moreover, the practical implementation of this strategy reveals a remarkable dependency on the number of parameters raw data points, in order to obtain an accurate resulting model. Furthermore, the parameter-dependent basis order chosen, together with the fitting points selected to the generation procedure, strongly affects the final result precision.

3.4.1 Numerical Results

In this section, we provide some numerical results that document the effectiveness of the above final stability enforcement. Nevertheless, all the examples reported in Chapter 7 will use this robust implementation as a standard for the stability enforcement and provides accurate results. The following cases are selected to stress the main aspect of the proposed algorithm. We refer to Test Cases of Appendix A for the structures under investigation.

Case 2
We now provide the numerical results from the same test case of Sec. 3.3.1, which is now passed to a 'predictive' final passivity enforcement. Table 3.4 demonstrates that the proposed robust implementation provides even better results in terms of model accuracy, while the overall time required for the model extraction remains invariant imposing only one 'predictive' iteration γ.

	Validation		Fitting	
	abs	rel	abs	rel
FPSK	8.61e-03	1.31e-02	7.47e-03	1.75e-02
FPSK with PR	1.01e-02	1.54e-02	1.16e-02	1.77e-02
FPSK with Predictive PR	1.01e-02	1.54e-02	1.16e-02	1.76e-02

Table 3.4: Maximum absolute and relative errors on validation and fitting points, computed over all the parameter values and port responses, for **Case 2** macromodel.

Case 3
We now investigate the same critical case of Sec. 3.3.1, which highlights a recursive perturbed denominator trend during the final passivity enforcement. Due to the advanced improvement provided above, a convergent solution is now reached.
By imposing a maximum number of predictive iterations $\gamma = 3$, the fitting algorithm requires only one denominator perturbation (μ) and succeeds obtaining a positive real admittance sub-model. The entire extraction requires only 38.6 seconds to succeed and provides a result with the same accuracy obtained without the robust implementation.

Indeed, Figure 3.6 compares the validation responses from the unstable model obtained after 20 passivity enforcement iterations on the denominator and the resulting model extracted with the robust strategy.
Around the same loss of accuracy is detected, which proves the efficacy of the 'prediction' iterations: it is shown that the robust implementation does not corrupt the model extraction. Moreover, we recall that the degradation of model accuracy is directly related to the restricted data-set, barely sufficient to provide a result. However, even if in this specific case the proposed approach does not completely solve the problem related to the fitting precision, which is affected by the problem itself (raw data available, basis functions order chosen and other secondary causes), the reduced number of final passivity enforcement iterations μ provides a general speed-up of the method. In particular, we support the above statements providing the maximum relative error for all the ports and parameters values as

- $e_{rel} = 5.53 \cdot 10^{-2}$ for a basic FPSK procedure;
- $e_{rel} = 2.10 \cdot 10^{-1}$ for the FPSK-scheme with a standard final passivity enforcement;
- $e_{rel} = 2.69 \cdot 10^{-1}$ for the FPSK with a robust implementation of the passivity enforcement.

Nevertheless, this is the only procedure available to guarantee a stable model extraction, as demonstrated in Fig. 3.6 (bottom): since $\mathrm{Re}\{\mathrm{D}(\mathrm{j}\omega, \vartheta)\} > 0$, it is guaranteed that the model poles are stable over $\vartheta \in \Theta$.

Moreover, we refer to the Test Cases of Appendix A where the effectiveness of the proposed guaranteed stable model extraction is proved, proving the maximum relative errors for all the examples used.

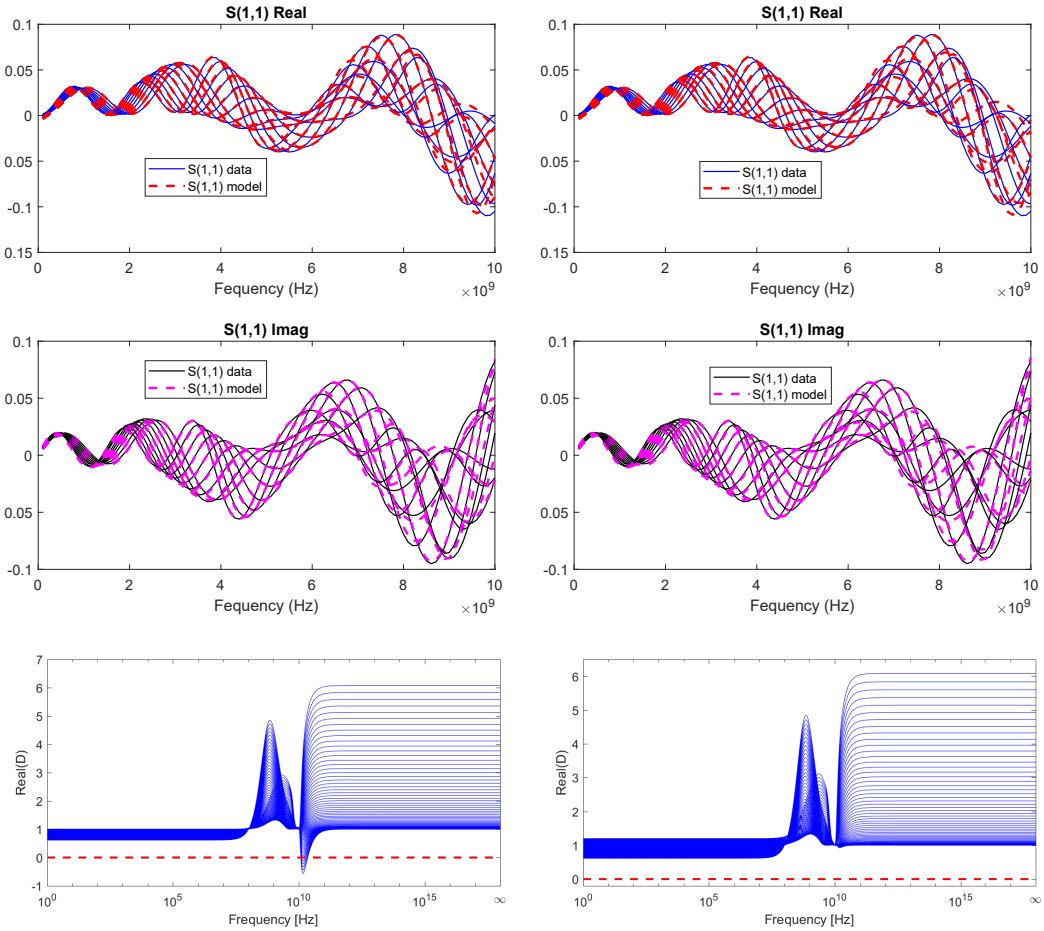


Figure 3.6: Validation of selected parameterized macromodel for **Case 3**. Left Panels, FPSK with denominator passivity enforcement, stopped at the 20-th iteration: (unstable) model responses and (bottom) real part of the model denominator, computed over a fine sweep of the parameter value.
Right Panels, FPSK with robust passivity enforcement: (stable) model responses and (bottom) real part of the model denominator.

Chapter 4

Equivalent circuit synthesis

Circuit solvers used in an industrial environment cannot always be interfaced with rational function-based models: usually, they only interpret netlists made of standard components, such as resistors, capacitors, inductors and controlled sources. Due to this limitation, user-provided models in the form (1.34) cannot be implemented in a circuit-level simulation, which is the main scope of our efforts. Therefore, we need to define a synthesis procedure to convert a rational macromodel in an equivalent circuit, which can be easily translated in a solver compatible netlist. In this context, literature offers several alternatives to convert models from state-space or pole residue form to an equivalent description that can be parsed by a circuit solver [19].

In the following sections, we focus on the state-space (sparse) synthesis, providing a process that allows the realization of an equivalent circuit, starting from the rational macromodel transfer function of (1.34). This procedure is not new [19,21], but we extend it to all the available input-output macromodel representations (scattering, admittance and impedance). Our work is always supported by numerical results: the function calls necessary to produce these examples are reported at the end of this chapter.

We place a word of caution here. A truncation error may be introduced in a numerical simulation, due to the restriction of the admissible number of digits in the definition of each circuit components [25]. This characteristic, depending on the circuit solver, may lead the overall system to a significant loss of passivity or stability. For this reason, it is suggested to define the equivalent circuit elements imposing at least the same number of significant digits available by the machine precision. In our examples, we always set an exponential notation of 20 digits with a precision of 16 digits.

4.1 Direct state-space synthesis

In order to synthesize an equivalent macromodel circuit, a direct formulation from its state-space description is a common procedure [3], [19]. Recalling the state-space equations

$$\begin{cases} \dot{\boldsymbol{x}}(t) = \mathbf{A}\boldsymbol{x}(t) + \mathbf{B}\boldsymbol{u}(t), \\ \boldsymbol{y}(t) = \mathbf{C}\boldsymbol{x}(t) + \mathbf{D}\boldsymbol{u}(t), \end{cases} \tag{4.1}$$

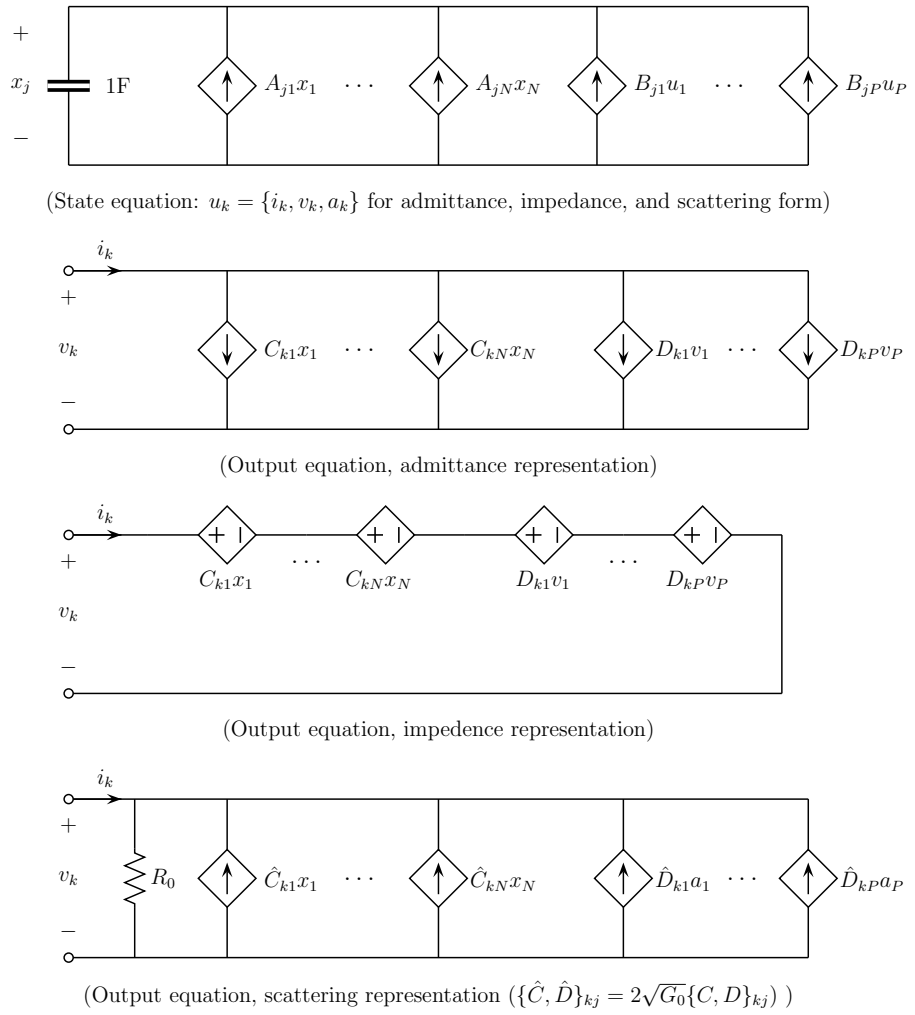


Figure 4.1: Direct circuit synthesis from state-space equations. Originally depicted in [19].

we can split their individual contributions as

$$\dot{x}_j(t) = \sum_{\ell=1}^{N} A_{j\ell} x_\ell(t) + \sum_{m=1}^{P} B_{jm}\, u_m(t), \tag{4.2}$$

$$y_k(t) = \sum_{\ell=1}^{N} C_{k\ell} x_\ell(t) + \sum_{m=1}^{P} D_{km}\, u_m(t). \tag{4.3}$$

An equivalent circuit representation of these equations is in Figure 4.1. The top row shows the first dynamic equation (4.2) where we assume to realize the state variable x_j as a voltage. The current that flows through the unitary capacitor corresponds to the state derivative, which is obtained as the sum of current sources: these are controlled by all the other states variables through the corresponding matrix elements $A_{j\ell}$, or by the input signals u_m and their coefficients B_{jm}. The macromodel representation changes

the synthesis of the output equation (4.3): the last three circuits of Figure 4.1 show the admittance, impedance and scattering representations. All of them take advantage of controlled sources. The first two realize a KCL and KVL, respectively. Instead the last one requires a mapping between circuit variables (i_k and v_k) and scattering variables (recalling that $2\sqrt{G_0}b_k = G_0 v_k - i_k$ where $G_0 = R_0^{-1}$). This is realized through the shunt resistance R_0 [19].

The procedure reported above (and presented in [19]), even if completely general and not dependent on the circuit solver, it is strongly affected by the large number of controlled source, which scales as $O(N^2P^2)$. As suggested in [19], we can try to exploit the arbitrariness of the state-space realization to produce a circuit equivalent either without controlled sources or with a minimum number of circuit components.
Both these solutions find their application field, such as noise analysis in RF and Mixed/Signal applications for the first one. Otherwise, reducing the number of circuit elements to a minimal quantity lead to an increased efficiency of circuit solver and, in general, to simulations speed-up.

4.1.1 Sparse synthesis

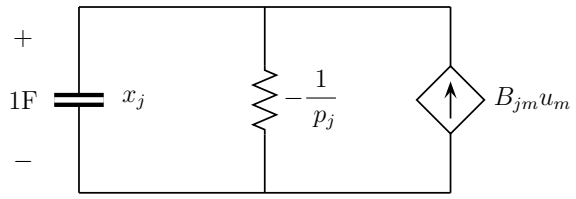


Figure 4.2: Contribution of a single real pole, for a direct circuit synthesis from state-space equations in the diagonal (multi-SIMO) realization case. Originally depicted in [19].

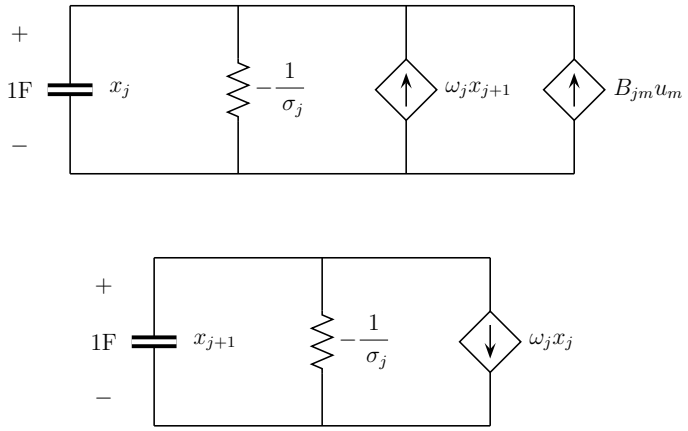


Figure 4.3: Contribution of a complex conjugate pole pair, same realization of Fig. 4.2. Originally depicted in [19].

Starting from a pole-residue model form

$$\mathbf{H}(s) = \mathbf{H}_\infty + \sum_{n=1}^{\bar{n}} \frac{\mathbf{R}_n}{s - q_n}, \tag{4.4}$$

we can realize a state-space macromodel equivalent description using the multi-SIMO (Single-Input Multi-Output) structure [19]. We define these systems as a number P of independent SIMO systems, each producing all P outputs, but each driven by a single independent input u_j. Furthermore, their state-space realization in matrix form is

$$\mathbf{A} = blkdiag\{\mathbf{A}_1, ..., \mathbf{A}_P\}, \qquad \mathbf{B} = blkdiag\{\boldsymbol{b}_1, ..., \boldsymbol{b}_P\}, \tag{4.5}$$

$$\mathbf{C} = (\mathbf{C}_1, ..., \mathbf{C}_P), \qquad \mathbf{D} = \{\boldsymbol{d}_1, ..., \boldsymbol{d}_P\}, \tag{4.6}$$

$$\tag{4.7}$$

where

$$\mathbf{A}_j = diag\{p_{j,1}, ..., p_{j,N_j}\}, \qquad \boldsymbol{b}_j = (1, ...,1)^T, \tag{4.8}$$

$$\mathbf{C}_j = diag\{\boldsymbol{r}_{j,1}, ..., \boldsymbol{r}_{j,N_j}\}, \qquad \boldsymbol{d}_j = \boldsymbol{r}_{j,0} \tag{4.9}$$

are the state-space matrices of the single j-th column realization $\boldsymbol{h}_j(s)$ for the model transfer function $\mathbf{H}(s)$ [19]. This description can be real if and only if all the N_j poles $\{p_{j,\ell}, l = 1, ..., N_j\}$ are real. Otherwise a blocks substitution of $\mathbf{A}_j$, $\boldsymbol{b}_j$ and $\mathbf{C}_j$ is required to obtain a purely real single-column: this allows to eliminate the imaginary contribution of the complex conjugate poles from the model residues $\boldsymbol{r}_{j,l}$. The procedure is

$$\begin{pmatrix} \sigma_{j,\ell} + j\omega_{j,\ell} & 0 \\ 0 & \sigma_{j,\ell} - j\omega_{j,\ell} \end{pmatrix} \leftarrow \begin{pmatrix} \sigma_{j,\ell} & \omega_{j,\ell} \\ -\omega_{j,\ell} & \sigma_{j,\ell} \end{pmatrix}, \tag{4.10}$$

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix} \leftarrow \begin{pmatrix} 2 \\ 0 \end{pmatrix}, \tag{4.11}$$

$$\begin{pmatrix} \boldsymbol{r}_{j,\ell} & \boldsymbol{r}^*_{j,\ell} \end{pmatrix} \leftarrow \begin{pmatrix} \mathrm{Re}\{\boldsymbol{r}_{j,\ell}\} & \mathrm{Im}\{\boldsymbol{r}_{j,\ell}\} \end{pmatrix}, \tag{4.12}$$

The single-lined contribution of the block-diagonal matrix $\mathbf{A}$ (which is necessarily composed by 2×2 blocks) corresponds to a real valued pole and it can be expressed as

$$\dot{x}_j(t) = A_{jj} x_j(t) + B_{jm} u_m(t) \tag{4.13}$$

for a generic m, with $A_{jj} = p_j < 0$ and $B_{jm} = 1$. A complex conjugate poles pair $p_j = \sigma_j \pm j\omega_j$ is, instead, embedded in two consecutive rows denoted as

$$\dot{x}_j(t) = \sigma_j x_j(t) + \omega_j x_{j+1}(t) + B_{jm} u_m(t) \tag{4.14}$$

$$\dot{x}_{j+1}(t) = -\omega_j x_j(t) + \sigma_j x_{j+1}(t) \tag{4.15}$$

for some m, with $\sigma_j < 0$ and $B_{jm} = 2$ (see [19]). Figure 4.2 and Figure 4.3 provide a circuit representation of (4.13) and (4.14) respectively, synthesising the diagonal entries of the matrix $\mathbf{A}$ with positive resistors and using controlled sources and capacitor for the state-space variables, following the same procedure adopted in Section 4.1.

Observing the complete solution of Figure 4.1(top), we can easily notice the greatest advantage of adopting a block-diagonal representation. In such case, considering the first dynamic equation (4.2), the total number of components range from $3N$ to $3.5N$ (assuming only real poles or only complex pair, respectively): to represent the dynamics through the state variable x_j only 3 circuit elements are required, which turn to 7 if we consider two states associated through a complex pole pair. On the other hand, using a full realization we can see that the number of components increases to $N^2 + N(P + 1)$, where $P + N + 1$ elements are necessary only for the j-th state x_j. The second (output) state-space dynamics equation usually remains the same: the three bottom representations of Figure 4.1 still hold, as well as (4.3). This is due to the fact that the residue matrices are normally full. In conclusion, the total number of circuit elements scales linearly with the definitions of the states, as $O(NP^2)$, for a sparse synthesis: this relation is instead quadratic in the complete case, with $O(N^2P^2)$.

4.2 An example

We now consider a two port capacitor (we refer to **Case 2** of Appendix A with sidelength fixed at 609.6 μm) with known reference scattering responses ($\bar{k} = 191$ frequency samples). This dataset was processed by proposed algorithm in order to obtain a stable and passive model with a corresponding SPICE netlist.

A good validation of the final model vs raw data is obtained using $\bar{n} = 4$ poles for numerator and denominator sub-models, imposing a relaxed normalization.

In these sections we are going to focus our attention on the admittance SPICE synthesis, providing numerical results. At first we will introduce a scalar case, using the denominator model sub-block previously presented. Then we will explore a multiport case, and in particular we will present a 2 x 2 port model using the numerator model sub-block as example.

4.2.1 Scalar Case

The state-space representation of the model denominator sub-block is the following

$$\mathbf{A}_D = \begin{pmatrix} q_1 & 0 & 0 & 0 \\ 0 & q_2 & 0 & 0 \\ 0 & 0 & \sigma_3 & \omega_3 \\ 0 & 0 & -\omega_3 & \sigma_3 \end{pmatrix} \qquad \mathbf{B}_D = \begin{pmatrix} 1 \\ 1 \\ 2 \\ 0 \end{pmatrix}, \tag{4.16}$$

$$\mathbf{C}_D = \begin{pmatrix} r_1 & r_2 & r_3 & r_4 \end{pmatrix} \qquad \mathbf{D}_D = \begin{pmatrix} r_0 \end{pmatrix} \tag{4.17}$$

where r_n are the gsk model coefficients of $\boldsymbol{r}$ (1.34).
The diagonal terms of $\mathbf{D}_D$ are realized as resistances in order to reduce the number of controlled sources inside the circuit. This results in a faster simulation. Moreover, this allows avoiding the critical case of having a netlist that could lead, in particular conditions, to a circuit configuration that does not present a DC path to ground.
We used a capacitance value $Cap = 1 \cdot 10^{-12}$. This optimal value is close to the inverse

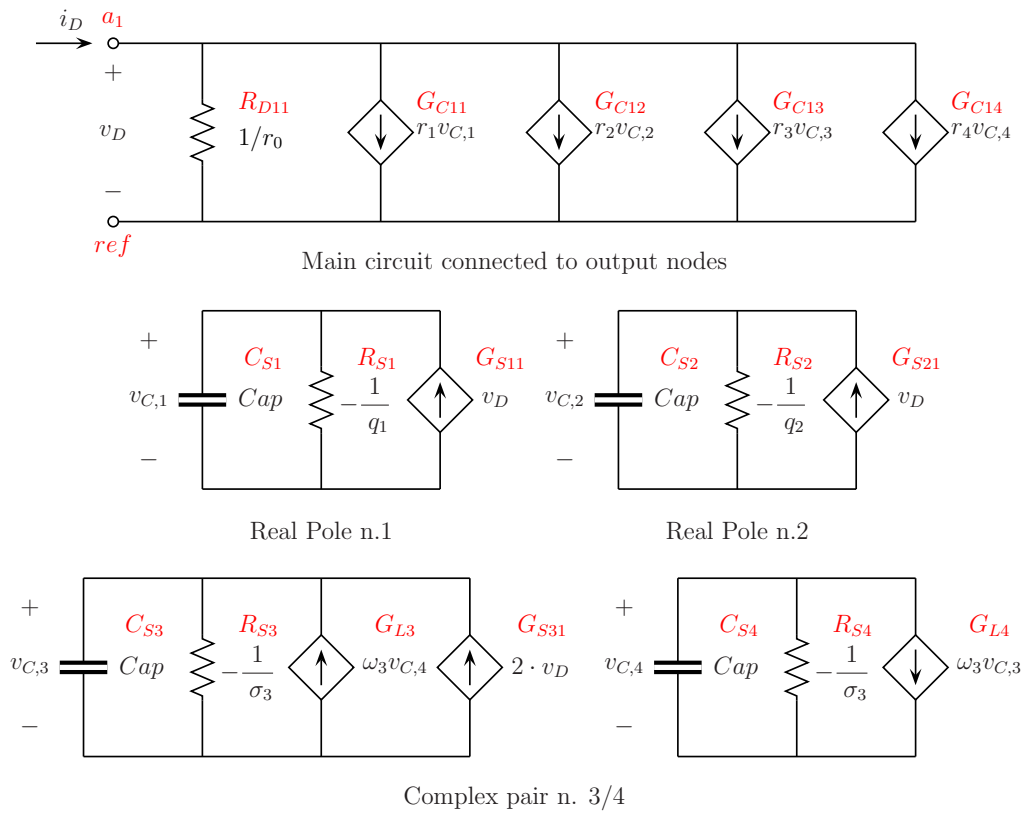


Figure 4.4: Synthesis of admittance block of the model denominator (netlist elements titled in red).

of the eigenvalues of the matrix $\mathbf{A}_D$. We only used the real part of the eigenvalues, by taking the average.

We realized a circuit synthesis imposing:

- a common reference node for all the ports;
- a standard resistor in the interface circuit.

The synthesis results in a SPICE netlist of the sub-circuit reported in the Figure 4.4. We can see the resulting netlist in the following script*[1]

```
****************************************
* SPICE subcircuit REALIZATION         *
* This file is automatically generated *
****************************************
* Created: 22-Dec-2017 by SS2Cir
****************************************
************************************************
```

[1]The note inserted inside the script as `[Text]` are not in a LTSpice compatible form: it should be removed to run the netlist.

```
* NOTE:
* a_i  --> input node associated to the port i
* ref  --> reference node, common for all input ports
**************************************************
**********************************
* Interface (ports specification) *
**********************************
.subckt GSKmodel_den
+  a_1 ref
**********************************
****************************************
* Main circuit connected to output nodes *
****************************************

* Port 1
GC_1_1 a_1 ref NS_1 0 2.6885991089072886e-01 [r1*vc1]
GC_1_2 a_1 ref NS_2 0 3.5761354482931279e-01 [r2*vc2]
GC_1_3 a_1 ref NS_3 0 9.2165830598959761e+00 [r3*vc3]
GC_1_4 a_1 ref NS_4 0 6.7301898754212051e+00 [r4*vc4]
RD_1_1 a_1 ref 3.6227104848874092e-02        [1/r0]
*
****************************************
**************************************
* Synthesis of real and complex poles *
**************************************

* Real pole n. 1
CS_1 NS_1 0 9.9999999999999998e-13           [Cap]
RS_1 NS_1 0 1.0072196180861735e+03         [-1/q1]
GS_1_1 0 NS_1 a_1 ref 1.0000000000000000e+00   [vd]
*
* Real pole n. 2
CS_2 NS_2 0 9.9999999999999998e-13           [Cap]
RS_2 NS_2 0 1.9664657708131415e+01         [-1/q2]
GS_2_1 0 NS_2 a_1 ref 1.0000000000000000e+00   [vd]
*
* Complex pair n. 3/4
CS_3 NS_3 0 9.9999999999999998e-13           [Cap]
CS_4 NS_4 0 9.9999999999999998e-13           [Cap]
RS_3 NS_3 0 2.7138694198861572e+01     [-1/sigma3]
RS_4 NS_4 0 2.7138694198861572e+01     [-1/sigma4]
GL_3 0 NS_3 NS_4 0 3.8108614844969030e-01  [w3*vc4]
GL_4 0 NS_4 NS_3 0 -3.8108614844969030e-01 [w3*vc3]
GS_3_1 0 NS_3 a_1 ref 2.0000000000000000e+00 [2*vd]
*
****************************
.ends
******************
* End of subcircuit
******************
```

4.2.2 Multiport Case

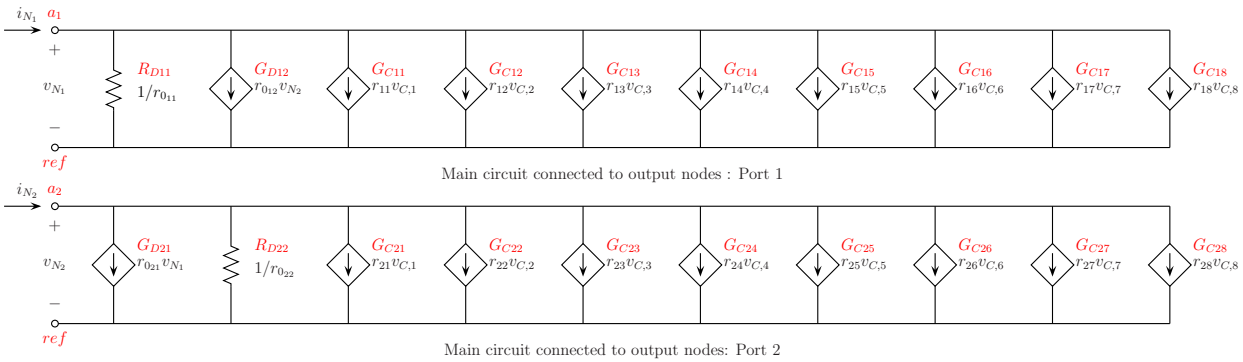


Figure 4.5: Synthesis of interface circuit of admittance block of the model numerator (netlist elements titled in red).

The state-space representation of the model numerator sub-block is the following

$$\mathbf{A}_N = \begin{pmatrix} \mathbf{A}_D & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_D \end{pmatrix} \qquad \mathbf{B}_N = \begin{pmatrix} \mathbf{B}_D & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_D \end{pmatrix}, \tag{4.18}$$

$$\mathbf{C}_N = \begin{pmatrix} r_{11} & r_{12} & r_{13} & r_{14} & r_{15} & r_{16} & r_{17} & r_{18} \\ r_{21} & r_{22} & r_{23} & r_{24} & r_{25} & r_{26} & r_{27} & r_{28} \end{pmatrix} \qquad \mathbf{D}_N = \begin{pmatrix} r_{0_{11}} & r_{0_{12}} \\ r_{0_{21}} & r_{0_{22}} \end{pmatrix} \tag{4.19}$$

where $r_{n,n}$ are the gsk model coefficients, elements of the matrix $\mathbf{R}_n$ (1.34). $\mathbf{A}_D$ and $\mathbf{B}_D$ are the matrices of the denominator state-space representation reported in sub-section 4.16. As it can be noticed, the two sub-blocks share the same poles.

We recall to the scalar example, of Section 4.16, for the capacitance value (*Cap*) choice. The circuit is realized imposing a common node for all the ports and a standard resistor, as in the scalar case.

The result is a SPICE netlist of the sub-circuit reported in Figure 4.5. The sub-circuits related to the poles are the same in Figure 4.4 but are repeated twice with increasing enumeration.

We can see the resulting netlist in the following script*[2]

```
****************************************
* SPICE subcircuit REALIZATION         *
* This file is automatically generated *
****************************************
* Created: 22-Dec-2017 by SS2Cir
****************************************


**************************************************
```

[2]The note inserted inside the script as `[Text]` are not in a LTSpice compatible form: it should be removed to run the netlist.

```
* NOTE:
* a_i  --> input node associated to the port i
* ref  --> reference node, common for all input ports
************************************************

**********************************
* Interface (ports specification) *
**********************************
.subckt GSKmodel_num
+  a_1 a_2 ref
**********************************

****************************************
* Main circuit connected to output nodes *
****************************************

* Port 1
GC_1_1 a_1 ref NS_1 0 2.7854871326954328e-01 [r11*vc1]
GC_1_2 a_1 ref NS_2 0 -5.1007306117455775e-01[r12*vc2]
GC_1_3 a_1 ref NS_3 0 -1.5357730431454714e-01[r13*vc3]
GC_1_4 a_1 ref NS_4 0 1.8314313421266970e-01 [r14*vc4]
GC_1_5 a_1 ref NS_5 0 -9.4560420360589124e-03[r15*vc5]
GC_1_6 a_1 ref NS_6 0 9.4023597990761154e-01 [r16*vc6]
GC_1_7 a_1 ref NS_7 0 1.1903241250691611e+01 [r17*vc7]
GC_1_8 a_1 ref NS_8 0 -1.5517141696119687e+00[r18*vc8]
RD_1_1 a_1 ref 1.8434241986958996e-01          [1/r011]
GD_1_2 a_1 ref a_2 ref -2.2838620218933343e+01[r012*vd2]
*
* Port 2
GC_2_1 a_2 ref NS_1 0 -9.4560420360589124e-03[r21*vc1]
GC_2_2 a_2 ref NS_2 0 9.4023597990761154e-01 [r22*vc2]
GC_2_3 a_2 ref NS_3 0 1.1903241250691611e+01 [r23*vc3]
GC_2_4 a_2 ref NS_4 0 -1.5517141696119687e+00[r24*vc4]
GC_2_5 a_2 ref NS_5 0 2.7917891062289341e-01 [r25*vc5]
GC_2_6 a_2 ref NS_6 0 -5.7478999133408770e-01[r26*vc6]
GC_2_7 a_2 ref NS_7 0 -9.7553525941326047e-01[r27*vc7]
GC_2_8 a_2 ref NS_8 0 3.0089649221146092e-01 [r28*vc8]
GD_2_1 a_2 ref a_1 ref -2.2838620218933343e+01  [r021*vd1]
RD_2_2 a_2 ref 1.4171794507218172e-01          [1/r022]
*
****************************************

************************************
* Synthesis of real and complex poles *
************************************

* Real pole n. 1
CS_1 NS_1 0 9.9999999999999998e-13
RS_1 NS_1 0 1.0072196180861735e+03
GS_1_1 0 NS_1 a_1 ref 1.0000000000000000e+00
```

```
*
* Real pole n. 2
CS_2 NS_2 0 9.9999999999999998e-13
RS_2 NS_2 0 1.9664657708131386e+01
GS_2_1 0 NS_2 a_1 ref 1.0000000000000000e+00
*
* Complex pair n. 3/4
CS_3 NS_3 0 9.9999999999999998e-13
CS_4 NS_4 0 9.9999999999999998e-13
RS_3 NS_3 0 2.7138694198861572e+01
RS_4 NS_4 0 2.7138694198861572e+01
GL_3 0 NS_3 NS_4 0 3.8108614844969030e-01
GL_4 0 NS_4 NS_3 0 -3.8108614844969030e-01
GS_3_1 0 NS_3 a_1 ref 2.0000000000000000e+00
*
* Real pole n. 5
CS_5 NS_5 0 9.9999999999999998e-13
RS_5 NS_5 0 1.0072196180861735e+03
GS_5_2 0 NS_5 a_2 ref 1.0000000000000000e+00
*
* Real pole n. 6
CS_6 NS_6 0 9.9999999999999998e-13
RS_6 NS_6 0 1.9664657708131386e+01
GS_6_2 0 NS_6 a_2 ref 1.0000000000000000e+00
*
* Complex pair n. 7/8
CS_7 NS_7 0 9.9999999999999998e-13
CS_8 NS_8 0 9.9999999999999998e-13
RS_7 NS_7 0 2.7138694198861572e+01
RS_8 NS_8 0 2.7138694198861572e+01
GL_7 0 NS_7 NS_8 0 3.8108614844969030e-01
GL_8 0 NS_8 NS_7 0 -3.8108614844969030e-01
GS_7_2 0 NS_7 a_2 ref 2.0000000000000000e+00
*
*******************************
.ends
********************
* End of subcircuit
********************
```

4.3 GSK Model Synthesis

We now consider the problem of synthesizing a standard (non-parameterized) GSK model as in (1.34) into a SPICE netlist, following and adapting the procedure of [21]. To avoid numerical issues with circuit solver simulations, we follow a strategy based on the conservation of the smooth parameterization of model coefficients, decomposing the model into separate interconnected blocks. The numerator and denominator model can be denoted as follow.

1. The denominator submodel is defined as a one-port admittance $Y_D(s) = \mathsf{D}(s)$, with auxiliary (dummy) port voltage and current v_D and i_D, respectively. We have

$$i_D = Y_D(s)\, v_D \quad \text{and} \quad v_D = Y_D^{-1}(s)\, i_D. \tag{4.20}$$

   The impedance $Z_D(s) = Y_D^{-1}(s) = \mathsf{D}^{-1}(s)$ is guaranteed (uniformly) stable, due to the PR enforcement embedded in the fitting algorithm (see Chapter 3). Moreover, we create P instances denominator, repeated identical and independent one for each other [See Fig. 4.6(b)]. The port voltages $v_{D,k}$ and currents $i_{D,k}$, for $k = 1, \ldots, P$, are gathered in vectors $\boldsymbol{v}_D, \boldsymbol{i}_D \in \mathbb{C}^P$. The result can be interpreted as

$$\boldsymbol{i}_D = \mathbf{D}(s)\, \boldsymbol{v}_D, \quad \boldsymbol{v}_D = \mathbf{D}^{-1}(s)\, \boldsymbol{i}_D, \tag{4.21}$$

   where $\mathbf{D}(s) = \mathrm{diag}\{\mathsf{D}(\mathrm{s})\}$.

   This procedure corresponds to the synthesis reported in Section 4.2.1, for a scalar case.

2. The numerator sub-model is defined as a P-port admittance matrix $\mathbf{Y}_N(s) = \mathbf{N}(s)$, with auxiliary (dummy) port voltage and current vectors $\boldsymbol{v}_N, \boldsymbol{i}_N \in \mathbb{C}^P$. We can set

$$\boldsymbol{i}_N = \mathbf{N}(s)\, \boldsymbol{v}_N, \tag{4.22}$$

   realized as in Fig. 4.6(a). This corresponds to the synthesis of the multiport case reported in Section 4.2.2.

3. The GSK model representation characterizes the synthesis of the last block, but the three cases present a common structure. Indeed we can write

$$\boldsymbol{y} = \mathbf{H}(s)\boldsymbol{u} = \mathbf{N}(s) \cdot \mathbf{D}^{-1}(s) \cdot \boldsymbol{u}. \tag{4.23}$$

   where $\boldsymbol{u}, \boldsymbol{y} \in \mathbb{C}^P$ are the input and output vectors, that depend by the chosen representation (scattering, admittance and impedance). The interconnection procedure between blocks is also common for the three forms, by setting

   - $\boldsymbol{v}_N = \boldsymbol{u}$, so that the current vector at the output of the numerator block reads $\boldsymbol{i}_D = \mathbf{N}(s)\, \boldsymbol{u}$. See Fig. 4.6(a), (scattering) and Fig. 4.7(a) (admittance and impedance);
   - $\boldsymbol{i}_D = \boldsymbol{i}_N$, so that $\boldsymbol{v}_D = \mathbf{H}(s)\, \boldsymbol{u}$. See elements (b) of Fig. 4.6 and Fig. 4.7;
   - $\boldsymbol{y} = \boldsymbol{v}_D$, so that $\boldsymbol{y} = \mathbf{H}(s)\, \boldsymbol{u}$. See Fig. 4.6(c), for scattering, and Fig. 4.7(c)(d), for admittance and impedance.

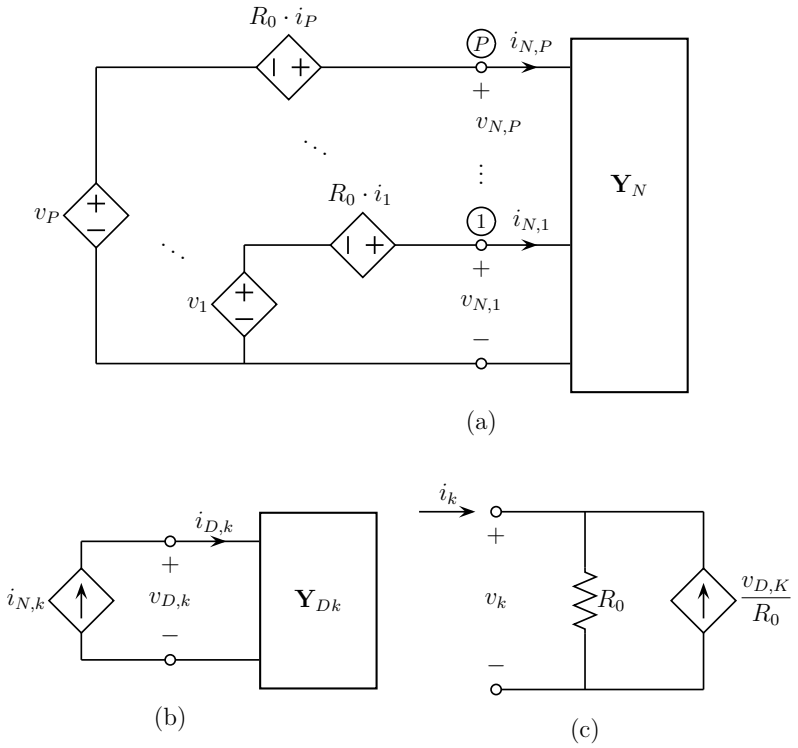


Figure 4.6: SPICE realization of (a) numerator sub-model; (b) denominator sub-model (k-th out of P instances); and (c) external interface of the GSK model (k-th out of P ports, scattering representation, realized in Norton form).

3.1. The model $\mathbf{H}(s)$ is here assumed to be in scattering representation. Therefore, the terms of equation (4.23) become,

$$\boldsymbol{y} = \boldsymbol{b} \quad \text{and} \quad \boldsymbol{u} = \boldsymbol{a}, \tag{4.24}$$

where $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{C}^P$ are the (voltage-normalized) incident and reflected scattering wave vectors, with components

$$a_k = (v_k + R_0 i_k), \tag{4.25}$$
$$b_k = (v_k - R_0 i_k), \tag{4.26}$$

and where R_0 is the port reference impedance. Interconnection of the various blocks is realized, as showed in Fig. 4.6 and detailed in the (general) procedure of point 3, by using

- a pair of controlled voltage sources to synthesize each incident wave a_k, as in the equation (4.25) (see Fig. 4.6(a));
- a set of current controlled current sources, as in Fig. 4.6(b);
- a current source, in parallel to the reference resistor R_0 to realize the output equation (4.26) in Norton form for each model port(see Fig. 4.6(c)).

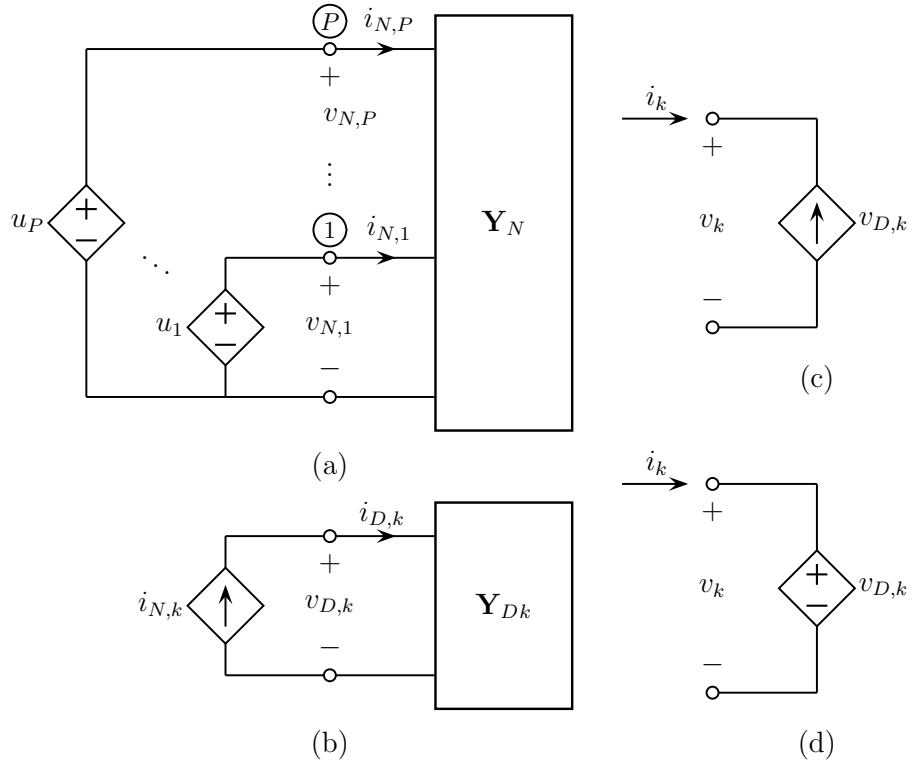


Figure 4.7: SPICE realization of (a) numerator sub-model in admittance and impedance form ($u_k = \{v_k; i_k\}$); (b) denominator sub-model (k-th out of P instances, common to the two representations); and external interface of the GSK model (k-th out of P ports) in (c) admittance or in (d) impedance form.

3.2. The model $\mathbf{H}(s)$ is here assumed to be in admittance representation. Therefore, the terms of equation (4.23) become,

$$\boldsymbol{y} = \boldsymbol{i} \quad \text{and} \quad \boldsymbol{u} = \boldsymbol{v}, \tag{4.27}$$

where $\boldsymbol{i}, \boldsymbol{v} \in \mathbb{C}^P$ are the input current and voltage vectors. Figure 4.7 shows interconnection of the blocks, following the steps detailed in point 3, by using

- a set of voltage controlled voltage sources (see Fig. 4.7(a), imposing $u_k = v_k$);
- a set of current controlled current source, as in Fig. 4.7(b);
- a set of voltage controlled current source, so that $\boldsymbol{i} = \mathbf{H}(s)\boldsymbol{v}$ (See Fig. 4.7(c)).

3.3. The model $\mathbf{H}(s)$ is here assumed to be in impedance representation. Therefore, the terms of equation (4.23) become,

$$\boldsymbol{y} = \boldsymbol{v} \quad \text{and} \quad \boldsymbol{u} = \boldsymbol{i}, \tag{4.28}$$

where $\boldsymbol{i}, \boldsymbol{v} \in \mathbb{C}^P$ are the same input current and voltage vectors of the admittance form. In this case, Fig. 4.7 shows interconnection of the blocks, again following the (general) procedure of point 3, by using

- a set of current controlled voltage sources (see Fig. 4.7(a), imposing $u_k = i_k$);
- a set of current controlled current source, as in Fig. 4.7(b);
- a set of voltage controlled voltage source, so that $\boldsymbol{v} = \mathbf{H}(s)\boldsymbol{i}$ (See Fig. 4.7(d)).

4.3.1 An example

Now we synthesize the example reported in Section 4.2, imposing a scattering representation. To this end we realize a wrapper circuit between the denominator and numerator admittance sub-blocks, previously presented as scalar (Section 4.2.1) and multiport example (Section 4.2.2).

The equivalent circuit is the one reported in Figure 4.6. We define $R_0 = 50\,\Omega$ as port reference resistance and a number of ports $P = 2$. We realize a circuit synthesis imposing:

- a common reference node for all the ports;
- a standard resistor in the interface circuit.

In this case the two files, containing the numerator and denominator admittance subcircuits, are included on top of the SPICE netlist.

We can see the result in the following script*[3]

```
****************************************
* SPICE subcircuit REALIZATION         *
* This file is automatically generated *
****************************************
* Created: 04-Dec-2017 by GSK_Model2Cir
****************************************

************************************************
* NOTE:
* a_i  --> input node associated to the port i
* ref  --> reference node, common for all input ports
************************************************

.INCLUDE GSKmodel_num.cir
.INCLUDE GSKmodel_den.cir

.subckt GSKmodel
+  a_1 a_2 ref

****************************************
* External/output ports of the network
****************************************
*Port: 1
G_e_1 ref tmp_e_1 c_d_1 0 0.02
```

[3] The note inserted inside the script as `[Text]` are not in a LTSpice compatible form: it should be removed to run the netlist.

```
R0_1 tmp_e_1 ref 50
Vtmp_e_1 a_1 tmp_e_1 0.0

*Port: 2
G_e_2 ref tmp_e_2 c_d_2 0 0.02
R0_2 tmp_e_2 ref 50
Vtmp_e_2 a_2 tmp_e_2 0.0

****************************************
* Numerator network (Cn)
****************************************
*Port: 1
E_1n_1 tmp1_n_1 0 a_1 ref 1.0
H_2n_1 tmp2_n_1 tmp1_n_1 Vtmp_e_1 50
Vtmp_n_1 tmp2_n_1 c_n_1 0.0

*Port: 2
E_1n_2 tmp1_n_2 0 a_2 ref 1.0
H_2n_2 tmp2_n_2 tmp1_n_2 Vtmp_e_2 50
Vtmp_n_2 tmp2_n_2 c_n_2 0.0

XNN_1 c_n_1 c_n_2 0 GSKmodel_num

****************************************
* Denominator network (Dn)
****************************************
*Port: 1
F_d_1 0 c_d_1 Vtmp_n_1 1
XDD_1 c_d_1 0 GSKmodel_den

*Port: 2
F_d_2 0 c_d_2 Vtmp_n_2 1
XDD_2 c_d_2 0 GSKmodel_den

.ends
```

4.4 Function Calls

In this part we report all the functions necessary to produce the synthesis of Chapter 4.
In particular, we will see the following functions:

- GSK_Model2Cir, driver to the other two functions;
- SS2Cir, realizes a sparse synthesis as in Section 4.1.1;
- MakeGSKWrapper, realizes a wrapper sub-circuit for the admittance sub-blocks, as described in Section 4.3.

4.4.1 GSK_Model2Cir

This function generates a SPICE sub-circuit that realizes a model in Generalized Sanathanan-Koerner form (non parameterized).

The input model must be a ratio between numerator and denominator series objects, which must be defined through the single-factor 'partialfractions' basis.

Options fields enables a customization of the SPICE synthesis, as detailed below. Here follows the function call.

```
function [NumSS,DenSS,om,Href] = GSK_Model2Cir(Model,pathname,name,
    Options)
```

Where the inputs are:

- Model is the model in the gsk form;
- pathname is the path where the output files will be located;
- name SPICE sub-circuit name in the output file *'name_.cir'* (Do not include the extension in the input string);
- Options is an (optional) input parameter that includes the fields
  - Options.GroundReferences [default = 0]
    determines how the reference nodes for all ports are generated. If it is set to 0, each port in the synthesized equivalent circuit will have a 'private' (floating) reference node. If it is set to 1, all ports will share a common reference node (useful for grounded multiports).
  - Options.ResistorType [default = 1]
    controls the synthesis of resistors in the equivalent circuit. These resistors are not 'true' resistors, but are just dummy components that are used to translate the model equations into a SPICE netlist. Therefore, these resistors might lead to wrong results when employed in a 'noise' analysis. Four different types of synthesis are available, according to the value of Options.ResistorType:
    1. synthesis as standard resistor (default)
    2. synthesis as a resistor with appended keyword 'noise=0' (available only for HSPICE)
    3. synthesis as current-controlled voltage source
    4. synthesis as voltage-controlled current-source

  - Options.mustOptimizeCap [default = 1]
    optimization of the capacitance value based on the location of the model poles (for GHz-range models, typical values are 1nF or 1pF). If false, all capacitances are set to 1F.
  - Options.debug [default = 0]
    if true, the function computes a validation of model response from individual numerator and denominator responses and returns the reference model response and angular frequency samples on output. Otherwise no validation is performed and both om and Href are empty.

On output, the function returns:

- NumSS, DenSS state-space realizations of numerator and denominator of input Model.
- om, Href angular frequencies and frequency response.

An example

To synthesize all the netlists used as examples in Chapter 4, it is necessary to type the following script[4]:

```
% path where files will be written
pathname = './testGSKModel_spice';
if ~exist(pathname,'dir'),
mkdir(pathname);
end

% name of Model netlist file
cktname = 'GSKmodel';

% netlist export options
Options.GroundReferences = 1;
Options.ResistorType = 1;
Options.mustOptimizeCap = 1;
Options.debug = 0;

GSK_Model2Cir(Model,pathname,cktname,Options);
```

4.4.2 SS2Cir

This function enables to synthesize a SPICE netlist of a sub-block in a (non-parametric) state-space form. It is assumed that state-space equations are a realization of a reduced-order model of a linear multiport, as specified in the sub-section 4.1.1.
The only dynamic elements included in the synthesis are identical capacitances. The

[4]The variable *Model* stores the corresponding element reported in Section 4.2, in a *IdEMPar* toolbox compatible structure.

capacitance value can be provided via field *Cap* of *MOD*. If this field is not present, a unitary value is used.

Here follows the function call

```
function SS2Cir(MOD, pathname, name, Options)
```

The inputs are:

- MOD variable that defines the state-space equations by fields MOD.A, MOD.B, MOD.C and MOD.D. The number of ports can be deduced by rows of C,D or columns of B,D. The field MOD.R0 is used to distinguish between admittance (MOD.R0 == 0) and scattering representation. In this second case the reference resistance used for all ports must be the same and it is also stored in MOD.R0. The capacitance value can be provided via field MOD.Cap. If this is not present, a unitary valued is used.
- pathname the path where the output files will be located
- name SPICE sub-circuit name in the output file *'name.cir'* (Do not include the extension in the input string)
- Options is an (optional) input parameter that includes the fields
  - Options.GroundReferences [default = 0] is an additional option that determines how the reference nodes for all ports are generated. If it is set to 0, each port in the synthesized equivalent circuit will have a 'private' (floating) reference node. If it is set to 1, all ports will share a common reference node (useful for grounded multiports).
  - Options.ResistorType [default = 1] controls the synthesis of resistors in the equivalent circuit. These resistors are not 'true' resistors, but are just dummy components that are used to translate the model equations into a SPICE netlist. Therefore, these resistors might lead to wrong results when employed in a 'noise' analysis. Four different types of synthesis are available, according to the value of Options.ResistorType:
    1. synthesis as standard resistor (default)
    2. synthesis as a resistor with appended keyword 'noise=0' (available only for HSPICE)
    3. synthesis as current-controlled voltage source
    4. synthesis as voltage-controlled current-source

4.4.3 MakeGSKWrapper

Constructs non-parameterized model main interface, consisting in a wrapper that connects the individual numerator and denominator netlists as detailed in Section 4.3.

Here is the function call

```
function makeGSKWrapper(R0, P, N_name, D_name, pathname, cktname, CommonGND)
```

The inputs are:

- R0 is the reference port resistance which allows to select the model representation (**0** admittance case ; **Inf** impedance case ; **0<R0<Inf** scattering case)

- P is the number of model ports
- N_name, D_name are the names of numerator and denominator sub-circuits, which are supposed to be available in individual files with same name and extension '*.cir'
- pathname is the path where the wrapper netlist will be located
- cktname is the name of the main sub-circuit and corresponding netlist file in the form 'name.cir'. (Do not include the extension in the input string)
- CommonGND is the type of interface port referencing scheme. If set to 0, each port in the synthesized circuit will have a 'private' (floating) reference node. If it is set to 1, all ports will share a common reference node.

Chapter 5

Parametric SPICE synthesis

We now consider the problem of synthesizing a parametric GSK model in a parameter-dependent SPICE netlist.

In this chapter we will explore a parametric SPICE-synthesis starting from the non-parametric procedure proposed in Chapter 4.3. In order to achieve this, we first introduce an overview of the new strategy, highlighting the main differences with the non-parametric synthesis procedure.

Then we focus our attention on the assignment of the parameter value to the GSK model sub-blocks, proposing three different call options. We detail each of them providing numerical examples. The chapter ends with the calls to the functions necessary to produce the parametric SPICE synthesis proposed.

5.1 Parametrized GSK Model Synthesis

The circuit synthesis strategy of a gsk-parametrized model is the same explored in Chapter 4.3. The resulting external interface circuit is similar to the one presented in Figure 4.6, depending on the choice made for the parameter call and synthesis.

The circuit realization of the parameterized admittance blocks corresponding to numerator and denominator is also standard [19, 20], as well as the proposed parameterization scheme [21]. We focus only to the synthesis of the denominator sub-model, which is a scalar: the extension to the (matrix) numerator case is straightforward. From (2.7), (2.8) and (4.21), we can denote

$$i_D = \mathsf{D}(s,\vartheta)\, v_D = \sum_{n=0}^{\bar{n}} j_{D,n} \tag{5.1}$$

where $j_{D,0} = r_0(\vartheta)\, v_D$ and

$$j_{D,n} = r_n(\vartheta)\, v_{C,n}, \quad \text{with} \quad v_{C,n} = (s - q_n)^{-1}\, v_D \tag{5.2}$$

for $n = 1, \ldots, \bar{n}$. An elementary RC circuit realizes each auxiliary variable $v_{C,n}$ (Fig. 5.1(a)). To simplify the notation we assume a set of real-valued[1] model poles $q_n < 0$.

[1] To realize a complex conjugate pair of poles, it is necessary to use two couple RC cells, see Fig. 4.3.

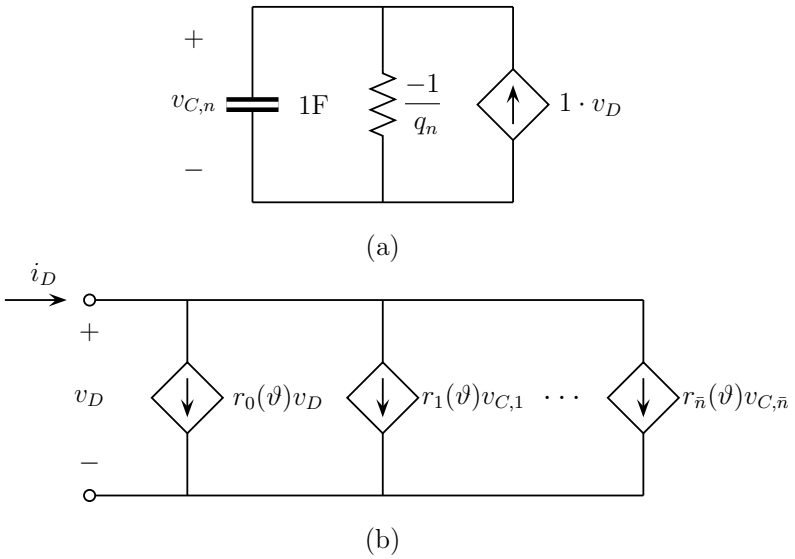


Figure 5.1: Realization of the parameter-dependent admittance block $\mathsf{D}(s, \vartheta)$ [21]: (a) elementary RC circuit that synthesizes the (real) basis pole q_n; (b) external interface circuit. ©2018 IEEE

We are going to expose in Chapter 6 different synthesis strategies to realize each auxiliary current $j_{D,n}$ using appropriate circuit elements.
In this case, a Voltage-Controlled Current Source (VCCS) is used for this purpose: the parameter-dependency is obtain through its trans-conductance $r_n(\vartheta)$. Such circuit elements, commonly avaible in the most used circuit solvers, allow to compute the overall current i_D of (5.1) as superposition of all these currents $j_{D,n}$, as showed in Fig. 5.1(b).

5.2 Parameter Call

Now we consider the problem of providing the parameter value to the model sub-circuit. This directly affects the GSK model interface netlist but with a slight modification of the parameter-dependent elements, as we will see in Chapter 6.
To achieve this purpose, we can follow three different strategies:

1. realize a **global parameter call**, defining a global variable in the simulation netlist which directly affects the parameter-dependent circuit elements in the admittance sub-blocks; as in Figure 5.2;

2. realize an **independent parameter call**, defining a variable in the simulation netlist and providing it as input to the macromodel sub-circuit; some local variables are defined in the wrapper netlist to provide the parameter as additional input to the two admittance sub-blocks, which contain the parameter-dependent elements; Figure 5.3 shows the parameter flow which allows for several variables;

3. realize a **Control Pin** call, providing the parameter values as voltage drop (or current flow) on an additional (control) input pin of the model sub-circuit; all the model

sub-blocks are instantiated with an additional pin and a cascade of connections enables the parameter-dependent elements to read the actual parameter value; Figure 5.4 shows a scheme of this strategy.

The first two methods require to create user-defined variables, that in LTSpice environment correspond to the `.param` directive. This way, sub-circuits can be parametrized and abstract circuits can be saved.
The parameters are passed to the sub-circuit by relations and expressions: using LTSpice, curly brackets are necessary to invoke parameter substitution and to evaluate an expression. The enclosed formulation is, in fact, reduced to a floating point numeric value, through the available relations, only when curly brackets are encountered (see LTSpice help for details [1]).

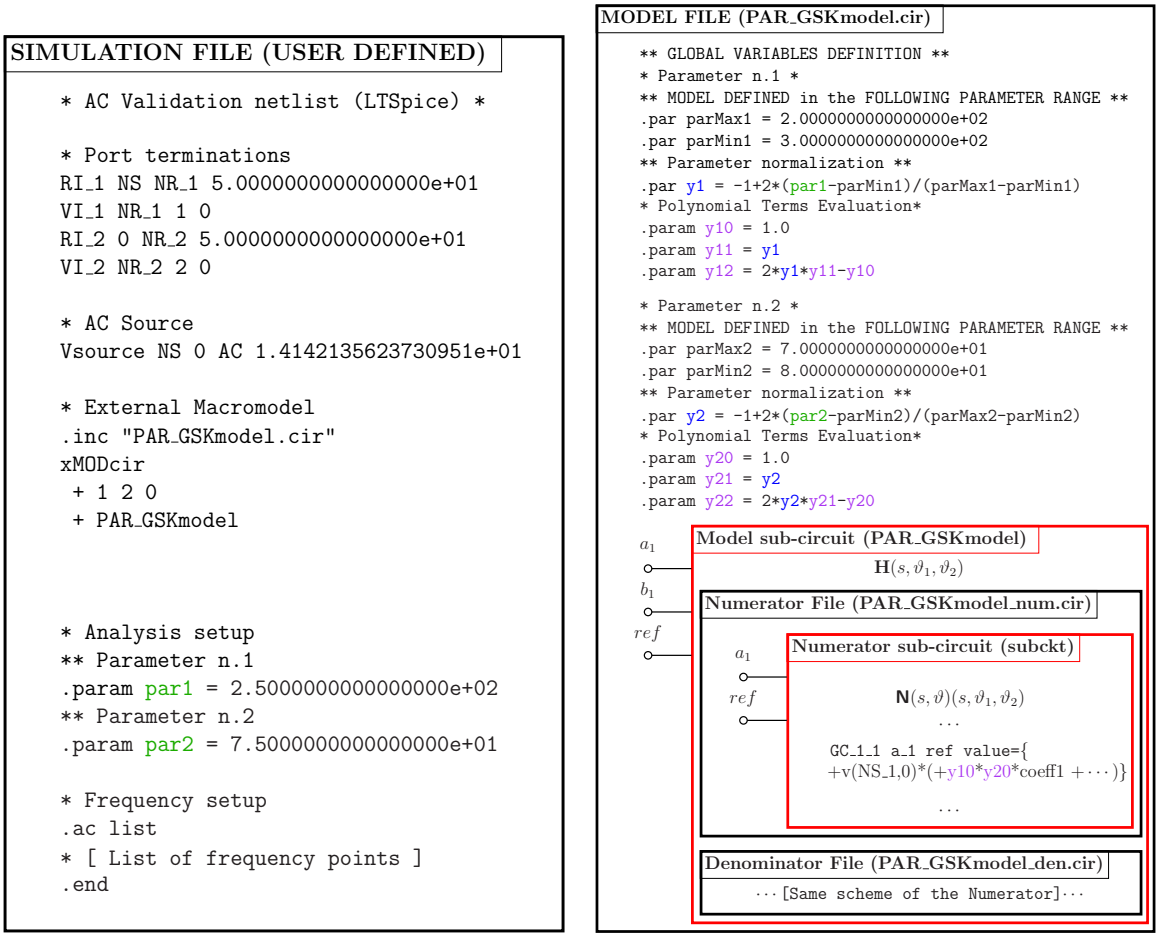


Figure 5.2: SPICE equivalent synthesis scheme of 2-port parametrized GSK-model adopting a global parameter call option. Boxes indicate either files (black) or sub-circuits (red). On the left, simulation file created by a user. On the right, model files generated by the automatic synthesis. Parameter flow is showed through colours.

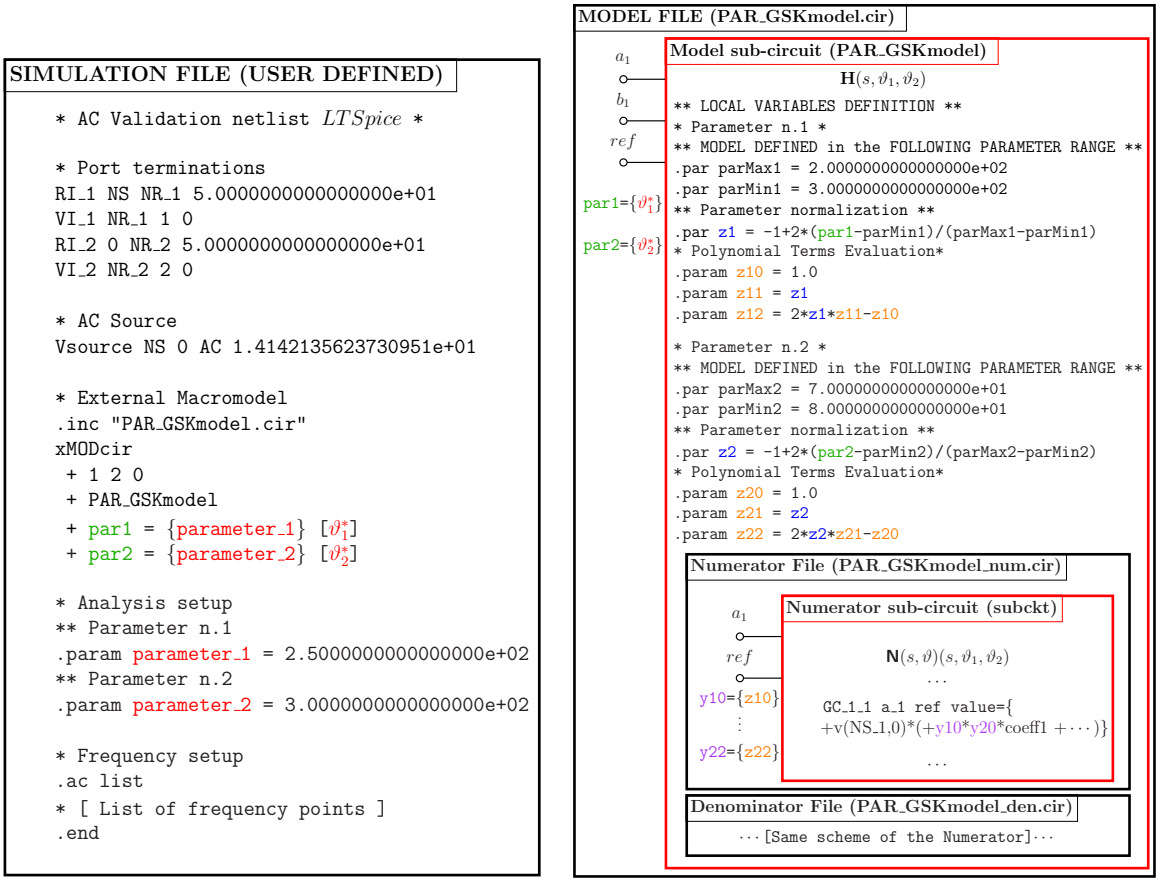


Figure 5.3: SPICE equivalent synthesis scheme of 2-port parametrized GSK-model adopting an independent parameter call option. Boxes indicate either files (black) or subcircuits (red). On the left, simulation file created by a user. On the right, model files generated by the automatic synthesis. Parameter flow is showed through colours.

We present now some common characteristics of the three strategies, that enable us to detail them in the next sections.

5.2.1 Parameter Normalization

For the synthesis of a multivariate macromodel, we used a particular class of orthonormal parameter-dependent basis known as Chebychev polynomials (see Section 2.1.1). To improve the numerical conditioning of fitting algorithms, as well as SPICE simulations, the range of the parameter variable is used to normalize the polynomial argument within [−1,1]. In particular, we compute the normalized parameter value $\tilde{\vartheta}$ as

$$\tilde{\vartheta} = -1 + 2 \cdot \frac{\vartheta - \vartheta_{min}}{\vartheta_{max} - \vartheta_{min}} \,. \tag{5.3}$$

where $\vartheta \in \Theta$ with $\Theta_i \in [\vartheta^i_{min}, \vartheta^i_{max}]$ is i-th parameter range, for $i = 1, ..., \rho$ (number of external design parameters).

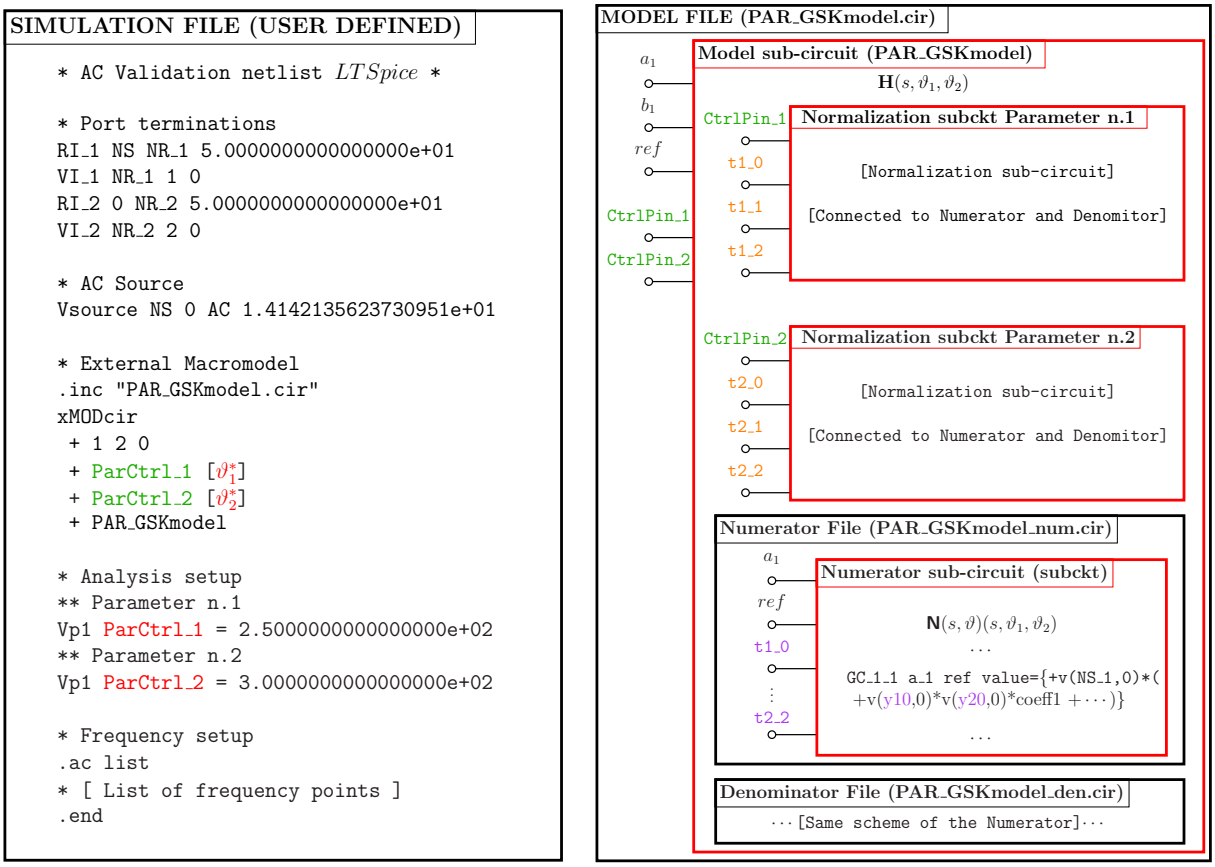


Figure 5.4: SPICE equivalent synthesis scheme of 2-port parametrized GSK-model adopting a control pin option. Boxes indicate either files (black) or sub-circuits (red). On the left, simulation file created by a user. On the right, model files generated by the automatic synthesis. Parameter flow is showed through colours.

In order to synthesize (5.3) in an equivalent expression to be parsed in a common circuit solver, we can follow two independent strategies:

1. realizing the operation directly as in the equation (5.3), defining user-defined variables compatible with the SPICE environment, through the LTSpice directive `.param`;
2. realize an equivalent normalization sub-circuit, which synthesizes an output voltage (or current) variable that matches the normalized parameter value.

The two strategies lead to the same identical results, taking to account possible truncation errors (see Chapter 4) of the element definitions, that may affect both variables or circuit components, in the normalized parameter computation.
Moreover, the above procedure must be repeated ρ times, one for each design parameter. The second case may lead to a loss of macromodel performances in numerical simulations, due to the increasing amount of circuit elements (usually controlled sources) with the number of design parameters.

5.2.2 Partial Evaluation of Parameter-Dependent Basis Functions

We report here a very important characteristic of the SPICE extraction procedure. The following observations are in fact crucial to an extension of the entire procedure for a generic parameter-dependent basis.

The parameter-dependent basis terms ξ_ℓ can be computed 'outside' the numerator or denominator admittances sub-blocks, where they are actually needed for the computation of the parameter-dependent circuit elements gain (i.e. the trans-conductance $r_n(\vartheta)$ of Fig. 5.1). In this case, they are provided to these sub-circuits as 'input' parameters.

This strategy presents three main advantages:

1. the specific choice of the parameter-dependent basis does not affect the admittance circuit elements, which preserve their parameter-dependency but now receive only the ξ_ℓ components as input;

2. the number of operations performed inside each sub-block is reduced;

3. the clarity of the netlist definition is increased even if the numerator and denominator parameter-dependent bases present different orders; this may reveal useful depending on the selected parameter call option.

We report now an example for the particular case of Chebychev polynomials: we will recall the following expressions for all the parameter calls in the next Sections.
Using the Three Term Recursive property (particular case of (2.14)), we can compute each term of ξ_ℓ as

$$\xi_0 = T_0 = 1 \tag{5.4}$$
$$\xi_1 = T_1 = \vartheta \tag{5.5}$$
$$\xi_\ell = T_\ell = 2\vartheta \cdot T_{\ell-1} - T_{\ell-2} \quad \text{for } \ell \geq 2\,. \tag{5.6}$$

We omit the i-th index on ϑ in order to simplify the notation: in a multivariate case, the above procedure must be repeated ρ times, one for each design parameter.

Note that, when the number of parameters increases, so does the overall amount of circuit components, and this may be lead to a slow-down of model performances in a numerical simulation.
However, the important advantages obtained by 'splitting' the parameter-dependent basis contributions overcome this hindrance and lead to realizing a flexible and performing netlist structure, also for a Control Pin call option and for a multivariate macromodel case.

In the next Sections, we will detail the three main parameter calls, providing examples.

5.3 Global Parameter

With this parameter-call choice, the parameter value is assigned directly to the transconductances inside admittance sub-blocks. All the input parameter definitions into the sub-circuits instances are omitted.
Choosing the proper variable name, a parameter value is automatically assigned to the circuit elements inside the admittance sub-circuits. We associated to the i-th parameter the netlist variable `par_i` , with $i = 1, ..., \rho$ (ρ number of additional parameters).

This is the most restrictive and dangerous case: two models with an identical parameterization cannot be instantiated together in a simulation, sharing the same parameter name and imposing two different values for this variable. The assignment difficulty grows in a multi-variate case: each variable must be identified with a unique name during the SPICE synthesis, and this can be done in the proper way only if the models which are going to be tested have a parameterization known *a priori*.
Even if this strategy for the parameter call allows to obtain the fastest simulations results, for the reason reported above it is affected by a loss of generality in more complex cases, resulting practically useless for an industrial environment.

Nevertheless, we will report this strategy because it will be the starting point for the other two implementations, providing basic but crucial concepts (and strategies) to reach the goal of a complete parametrized synthesis.

5.3.1 Parameter in Wrapper and Admittance Sub-circuits

Contrary to what one would expect, nothing changes during the synthesis of the PSK-model wrapper netlist with a **global parameter** call: this strategy modifies directly the circuit elements definitions inside each admittance sub-block (see example Section 5.1), maintaining their instances invariant.
Assigning the desired parameter value to a variable in the simulation netlist, this is automatically cast to each circuital component, without the necessity of specifying any kind of relationship with the parametrized-macromodel sub-block.
The only adjustments required by selecting this option are related to

1. the parameter normalization
   We realize the first procedure described in Section 5.2.1 as[2]:

```
** Parameter normalization **
.param parMax_i = thetaMax_i
.param parMin_i = thetaMin_i
.param y_i = -1+2*(par_i-parMin_i)/(parMax_i-parMin_i)
```

   where `par_i` and `y_i` correspond respectively to $\vartheta^i \in \Theta_i$ and $\tilde{\vartheta}^i$ of (5.3). Both of them are user-defined variables in the LTSpice environment, as well as `parMax_i` and

[2] `thetaMax_i` and `thetaMin_i` should be substituted with their numerical values, corresponding to $[\vartheta^i_{min}, \vartheta^i_{max}] = \vartheta_i$, where $i = 1, ..., \rho$. Moreover, the same index i should substitute the 'netlist index' `*_i`.

`parMin_i`. We recall that, in a multivariate model case, this code fragment must be repeated for all the design parameters, substituting the index `*_i` in the netlist with a number, according to the index of ϑ^i.

2. the evaluation of the paramereter-dependent basis terms $\xi_\ell(\vartheta)$
   Recalling to Sec. 5.2.2 and following again the procedure that requires user-defined variables, we define a SPICE equivalent of (5.4) as

```
** Polynomial Terms Evaluation **
.param y_i0 = 1.0
.param y_i1 = y_i
.param y_i2 = 2*y_i*y_i1-y_i0
```

   which is the realization of the first three terms of the parametric-dependent basis(`y_i0` , `y_i1` and `y_i2` respectively), for the particular case of the Chebychev polynomial. The extension of this procedure to any general basis is straightforward. Moreover, the multivariate case follows the same rules denoted above for the parameter normalization.

Both these procedures must be performed in the GSK-model file: this is a key point for this SPICE equivalent realization.
It is crucial, for this strategy, that these two points are realized *outside* the wrapper sub-circuit definition. Otherwise, the parameter-dependent variables defined in the wrapper netlist will not be read by the internal sub-circuits, which in this case are exactly the numerator and denominator admittance sub-blocks (see Figure 5.2).

5.3.2 An Example

Now we are going to provide an example of the global parameter call: the following initial set-up will be reused for the other two strategies, quoted in Section 5, to stress the differences between the three parameter calls options available. In order to achieve this, we will synthesize each model for each different procedure, omitting all the netlist parts that are not relevant at this realization level. In particular, we will explore the admittance sub-blocks synthesis in the Chapter 6.
To guarantee a complete overview of the parameter flow inside the model equivalent, we realized a unique netlist merging together the GSK-model and the simulation file: the change of source code in the examples is detailed with a comment (`*[File]`).

For this purpose we used a $P = 2$ ports model, with only one external parameter ($\rho = 1$) and a parameter-dependent basis cardinality for the numerator of $\ell_N = 4$ and for the denominator of $\ell_D = 3$. The frequency basis order is omitted. The parameter is defined as $\vartheta \in [2{,}18]$. The parameter basis terms are evaluated only once and passed to the admittance sub-blocks, as discussed in Section 5.3.1.

We recall now the global parameter call key points, which can be observed in the example reported below.
In this case, no specification related to the parametric nature of the model is presented

in any sub-circuit call or definition.
A variable, necessary to store the parameter value, must be defined in the simulation circuit only. The name of this variable and the one chosen during the global variable synthesis must match. No curly brackets invocation is required at this level: the parameter-dependent circuit elements of the admittance sub-blocks directly receive the parameter value. Moreover, the parameter-dependent basis terms are computed in a global environment only once, just before the GSK-model sub-circuit definition.

Here follows the SPICE netlist[3].

```
**************************************
**[SIMULATION FILE (Sim.cir)]**
**************************************
* AC Validation netlist (LTSpice)      *
************************
* Port terminations
[...Chosen Simulation Set Up...]

************************
* Parameters setup (User defined)
************************

* Parameter n.1
.param par1 = 2.0000000000000000e+00

************************
* External Macromodel
************************

.inc 'PAR_GSKmodel.cir'
xMODcir
+  1 2 0
+ PAR_GSKmodel

**************************************
**[GSK-MODEL FILE (Model.cir)]**
**************************************

*******************************
** GLOBAL VARIABLES DEFINITION **

** MODEL DEFINED in the FOLLOWING PARAMETER RANGE **
.par parMax1 = 1.8000000000000000e+01
.par parMin1 = 2.0000000000000000e+00

************************
** Parameter normalization **
```

[3] Text inserted inside the script as `[Text]` it should be substituted with the specified subject, in a LTSpice compatible form.

```
************************

.par y1 = -1+2*(par1-parMin1)/(parMax1-parMin1)

************************
** Polynomial Terms Evaluation **
************************

.param y10 = 1.0
.param y11 = y1
.param y12 = 2*y1*y11-y10
.param y13 = 2*y*y12-y11
.param y14 = 2*y*y13-y12

************************
** end of GLOBAL VARIABLES DEFINITION **
*******************************

.INCLUDE PAR_GSKmodel_num.cir
.INCLUDE PAR_GSKmodel_den.cir

************************
*** BEGIN: parameterized macromodel
************************
.subckt PAR_GSKmodel

** Macromodel Ports Definition **
+  a_1 a_2 ref

************************
* NOTE:
* a_i  --> input node associated to the port i
* ref  --> reference node, common for all input ports
************************

************************
* External/output ports of the network
************************
[...Chosen Representation...]

************************
* Numerator network (Cn)
************************
[...Chosen Representation...]
XNN_1 c_n_1 c_n_2 0 PAR_GSKmodel_num

************************
* Denominator network (Dn)
************************
*Port: 1
```

```
F_d_1 0 c_d_1 Vtmp_n_1 1
XDD_1 c_d_1 0 PAR_GSKmodel_den

*Port: 2
F_d_2 0 c_d_2 Vtmp_n_2 1
XDD_2 c_d_2 0 PAR_GSKmodel_den

.ends * END parameterized macromodel

****************************************
**[SIMULATION FILE (Sim.cir)]**
****************************************
************************
* Frequency Analysis setup
************************
.ac list
[...Frequency Points...]

.end * END simulation
```

5.4 Independent Parameter

The parameters are defined as additional input variables of each sub-circuit. This characteristic enables a direct control on the parameter name definition by the user, which corresponds to a better synthesis strategy flexibility.

With this call option, the user can provide the chosen parameter value *trough* the input variable denoted as `par_i` , with $i = 1, ..., \rho$. Such netlist variables are embedded in the wrapper sub-circuit definition and are initialized to 1 (default value).

The main difference with the global parameter call from a user point-of-view is exactly the necessity to provide the parameter value as additional input to the macromodel instance, using curly bracket invocation and the proper variable name.
To clarify this point, we provide an example for a multivariate model with two external parameters, according to the LTSpice environment[4]:

```
.subckt Parametric_GSKmodel
+  a_1 a_2 ref
+  params: par1 = 1
+  params: par2 = 1
[... Circuit Definition ...]
.ends Parametric_GSKmodel
```

The previous script requires to call a variable exactly as `par_i` (with $i = 1,2$) in the sub-circuit invocation line, using curly brackets to assign a value to the i-th parameter. If we want to fix their quantities to $\vartheta_1 = 3$ and $\vartheta_2 = 7$, the following lines must be used:

```
.include 'Parametric_GSKmodel.cir'
xModelCir
+  Pin1 Pin2 ref
+ Parametric_GSKmodel
+  par1 = {3}
+  par2 = {7}
```

The information flow, required to satisfy this sub-block definition, implies a slight loss of model performances with respect to the **global parameter** call strategy, which we verified that does not compromise the overall simulation results. On the other hand, this second strategy allows to overcome all the drawbacks presented by the first one (see Section 5.3).

We now detail the parameter role inside the wrapper and admittance sub-blocks and the modifications required to realize these two sub-circuits compatible with this parameter call strategy.

[4]Text inserted inside the script as `[Text]` should be substituted with the specified subject, in a SPICE compatible form.

5.4.1 Parameter in Wrapper and Admittance Sub-circuits

We previously described how the parameters are provided to the wrapper sub-block: in this case, the Independent parameter call modifies also the instance of this element, as well as the admittance sub-circuits definitions.
Another important characteristic of this procedure is the definition of 'local' variables *inside* the GSK-model sub-block, which are responsible for the parameter normalization and for the partial evaluation of the parameter-dependent basis. These internal variables are provided as additional input to the admittance sub-blocks, following the same procedure reported for the GSK-model interface sub-circuit.
We now extend the details of the same two key points of Section 5.3.1 concerning the global parameter option, with the modifications required for this case.

1. parameter normalization.
   Changing the name of the variable associated to the i-th normalized parameter $\tilde{\vartheta}^i$ (previously called `y_i`), the same procedure reported for the global parameter call is realized as[5]:

   ```
   ** Parameter normalization **
   .param parMax_i = thetaMax_i
   .param parMin_i = thetaMin_i
   .param z_i = -1+2*(par_i-parMin_i)/(parMax_i-parMin_i)
   ```

   where `par_i` and `z_i` correspond respectively to $\vartheta^i \in \Theta_i$ and $\tilde{\vartheta}^i$ of (5.3).

2. evaluation of the paramereter-dependent basis terms $\xi_\ell(\vartheta)$.
   Also in this case, the variables names must be changed accordingly to point 1 but the procedure is the same of Section 5.3.1

   ```
   ** Polynomial Terms Evaluation **
   .param z_i0 = 1.0
   .param z_i1 = z_i
   .param z_i2 = 2*z_i*z_i1-z_i0
   ```

   where the first three terms of the Chebychev polynomial basis are realized for the i-th parameter. The same observations made for the global parameter call still hold.

These modifications are necessary because, during the SPICE equivalent synthesis, we associated `y_il` to the variable which 'lives' inside the parameter-dependent circuit elements (i.e. it contributes to the definition of the parameteric-dependent transconductances), as it can be seen in Fig.5.3.
These local variables are than provided as additional input to the admittance sub-blocks as in the following netlist example:

```
XNum 1 2 0 PAR_GSKmodel_num
+ yi0 = {z_i0} y_i1 = {z_i1} y_i2 = {z_i2}
```

[5] `thetaMax_i` and `thetaMin_i` should be substituted with their numerical values, corresponding to $[\vartheta^i_{min}, \vartheta^i_{max}] = \vartheta_i$, where $i = 1, ..., \rho$. Moreover, the same index i should substitute the 'netlist index' `*_i`.

where `PAR_GSKmodel_num` is the numerator sub-model of a 2×2 ports system (with common reference node imposed during its synthesis, see Chapter 4) and the `y_il` variables are the one reported before.

We recall that, with this parameter call strategy, the two points above must be realized ρ times (with ρ external parameters) *inside* the wrapper sub-circuit definition. Figure 5.3 shows a schematic of this procedure.

5.4.2 An Example

We now provide an example of a model synthesis through an independent parameter call, using the same set-up of Section 5.3.2.
In this case, parameter definition and invocation are unambiguous. The variable associated with the parameter must be provided as input of the macromodel by inserting inside curly brackets the appropriate parameter name, as specified in the section `NOTE` of the interface sub-circuit netlist reported below.

Here follows an example netlist[6].

```
****************************************
**[SIMULATION FILE (Sim.cir)]**
****************************************
* AC Validation netlist (LTSpice)      *
************************
* Port terminations
[...Chosen Simulation Set Up...]

************************
* Parameters setup (User defined)
************************

** Parameter n. 1
.param parameter1 = 2.0000000000000000e+00

****************************************
* External Macromodel
****************************************

.inc 'PAR_GSKmodel.cir'
xMODcir
+  1 2 0
+ PAR_GSKmodel
+ par1 = {parameter1}

****************************************
```

[6] Parts inserted inside the script as `[Text]` should be substituted with the specified subject, in a LTSpice compatible form.

```
**[GSK-MODEL FILE (Model.cir)]**
****************************************

.INCLUDE PAR_GSKmodel_num.cir
.INCLUDE PAR_GSKmodel_den.cir

************************
*** BEGIN: parameterized macromodel
************************
.subckt PAR_GSKmodel

** Macromodel Ports/Parameters Definition **
+  a_1 a_2 ref
+  params: par1 = 1

************************
* NOTE:
* a_i  --> input node associated to the port i
* ref  --> reference node, common for all input ports
* par_i --> i-th parameter value in input to the netlist
************************

*******************************
** LOCAL VARIABLES DEFINITION **

** MODEL DEFINED in the FOLLOWING PARAMETER RANGE **
.par parMax1 = 1.8000000000000000e+01
.par parMin1 = 2.0000000000000000e+00

************************
** Parameter normalization **
************************

.par z1 = -1+2*(par1-parMin1)/(parMax1-parMin1)

************************
** Polynomial Terms Evaluation **
************************

.param z10 = 1.0
.param z11 = z1
.param z12 = 2*z1*z11-z10
.param z13 = 2*z1*z12-z11
.param z14 = 2*z1*z13-z12

************************
** end of LOCAL VARIABLES DEFINITION **
*******************************

************************
```

```
* External/output ports of the network
************************
[...Chosen Representation...]

************************
* Numerator network (Cn)
************************
[...Chosen Representation...]

XNN_1 c_n_1 c_n_2 0 PAR_GSKmodel_num
+ y10 = {z10} y11 = {z11} y12 = {z12} y13 = {z13} y14 = {z14}

************************
* Denominator network (Dn)
************************
*Port: 1
F_d_1 0 c_d_1 Vtmp_n_1 1
XDD_1 c_d_1 0 PAR_GSKmodel_den
+ y10 = {z10} y11 = {z11} y12 = {z12} y13 = {z13}

*Port: 2
F_d_2 0 c_d_2 Vtmp_n_2 1
XDD_2 c_d_2 0 PAR_GSKmodel_den
+ y10 = {z10} y11 = {z11} y12 = {z12} y13 = {z13}

.ends * END parameterized macromodel

**************************************
**[SIMULATION FILE (Sim.cir)]**
**************************************
************************
* Frequency Analysis setup
************************
.ac list
[...Frequency Points...]

.end * END simulation
```

5.5 Control Pin

If a **control pin** interface is adopted, then an additional pin is added to the parametrized GSK-model sub-circuit definition. Here follows an example for a 2 ports model with two external parameters $\vartheta = (\vartheta_1, \vartheta_2)$:

```
.subckt Parametric_GSKmodel
+  a_1 a_2 ref
+ CtrlPin_1
+ CtrlPin_2
[... Circuit Definition ...]
.ends Parametric_GSKmodel
```

By connecting a voltage source to the control pins, defined as `CtrlPin_1` and `CtrlPin_2`, in the caller netlist, the requested parameter values are correctly provided to the model circuit. Using the same example presented for the independent parameter call case (see Section 5.4), the resulting netlist is:

```
.include 'Parametric_GSKmodel.cir'
xModelCir
+  Pin1 Pin2 ref
+ ParCtrl_1
+ ParCtrl_2
+ Parametric_GSKmodel
Vpar1 ParCtrl_1 0 3
Vpar2 ParCtrl_2 0 7
```

where `Vpar1` and `Vpar2` are standard voltage sources that provide the parameters values as voltage drops on the corresponding model (control) input pins.

We now detail, as we have done for the other strategies, the parameter role in the wrapper and admittance sub-blocks: a new procedure must be used, which avoids user-defined variables.

5.5.1 Parameter in Wrapper and Admittance Sub-circuits

As for the independent parameter case (Sec. 5.4.1), also when a control pin is adopted all the model sub-circuits instances must be modified according to the procedure reported above: this still holds for the wrapper and for the admittance sub-blocks.

The main characteristic that distinguishes this choice from other strategies is the following.
When an additional pin is used to control the parameter evaluation, a user-defined variable cannot be used to store the input value. For this reason, the code presented for the other two procedures (Sec. 5.3.1 and Sec. 5.4.1) is replaced with specific sub-circuits that perform the same computations.
The critical points are again related to the

1. parameter normalization.

In LTSpice, a controlled source can be used to read the voltage drop on the input control pin and at the same time to perform the computation seen in (5.3). We can define a 'normalization' sub-circuit (`ParNormalization`) as[7]:

```
.subckt ParNormalization CtrlPin_i T_i_1
RCtrlP_i CtrlPin_i 0 1e9
ET_i_1 T_i_1 0 value={-1+2*(v(CtrlPin_i,0)-thetaMin_i)/
+ (thetaMax_i-thetaMin_i)}
RT_i_1 T_i_1 0 1e9
.ends ParNormalization
```

The actual i-th parameter value is provided as input voltage drop on the control pin (`CtrlPin_i`).

Therefore, a Voltage Controlled Voltage Source (VCVS) is used to impose the normalized parameter value as the voltage drop on the output pin (`T_i_1`), which is connected to the two admittance sub-blocks. The VCVS gain corresponds exactly to the value stored in the variable `z_i` used in the Section 5.4.1: the two procedures are completely equivalent.

Two grounded resistors, wired to the sub-circuit interface pins, are necessary because some spice implementations may not allow floating nodes. Resistances values are set to 1 GΩ to reduce the absorbed power from the Control Pin.

2. evaluation of the paramereter-dependent basis terms $\xi_\ell(\vartheta)$.

   In order to simplify the notation of the netlist example, in the following, we will consider only one external parameter ($\rho = 1$). We can extend the procedure to a multivariate case adding the index `*_i` , as in the normalization circuit above.

   A number of $2 \cdot \ell$ circuit elements must be added to the normalization sub-circuit, where ℓ is the maximum parameter basis cardinality between the numerator and denominator sub-models.

   The new 'normalization' circuit presents now $\ell + 1$ pins: the first must be wired to the model Control Pin, while the others are internally connected to the parameter-basis terms sources and return the corresponding basis function values. Each one of these is computed using a VCCS element, directly connected to the corresponding output pin and to the ground.
   A number $(\ell + 1)$ of grounded resistors are necessary because some SPICE implementations may not allow floating nodes.

   Here we present an example, creating a sub-circuit to compute the first four terms of the Chebychev polynomial basis ($\ell = 4$):

```
.subckt cheb CtrlPin T_0 T_1 T_2 T_3
RCtrP CtrlPin 0 1e9
VT0 T_0 0 1
```

[7] `thetaMax_i` and `thetaMin_i` should be substituted with their numerical values, corresponding to $[\vartheta^i_{min}, \vartheta^i_{max}] = \vartheta_i$, where $i = 1, ..., \rho$. Moreover, the same index i should substitute the 'netlist index' `*_i`.

```
RT0 T_0 0 1e9
ET1 T_1 0 value={-1+2*(v(CtrlPin,0)-thetaMin)/
    + (thetaMax-thetaMin)}
RT1 T_1 0 1e9
ET2 T_2 0 value={2*v(T_1,0)*v(T_1,0)-v(T_0,0)}
RT2 T_2 0 1e9
ET3 T_3 0 value={2*v(T_1,0)*v(T_2,0)-v(T_1,0)}
RT3 T_3 0 1e9
.ends ParNormalization
```

where the parameter normalization is performed by the second polynomial term realization, through the definition of `ET1`.
A constant voltage source is used to compute the first Chebychev element in order to reduce the overall number of controlled components, according to the observations of Chapter 4.

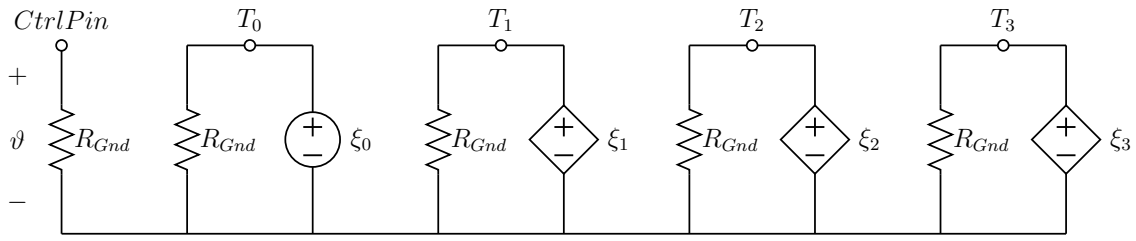


Figure 5.5: Synthesis of the normalization sub-circuit `cheb` necessary to evaluate and return each term of the Chebychev basis as output voltage drop on the corresponding pin. The grounded resistors R_{Gnd}, set to 1 GΩ, are necessary because some SPICE implementations may not allow floating nodes. Only one parameter is considered.

In the GSK-model netlist, the sub-circuit presented (`cheb`) must be created only once (for each i-th parameter) and will be connected to the numerator and denominator sub-blocks though the same out pins.
Figure 5.4 shows a general scheme of the above procedure for a multivariate model, with two external parameters.

5.5.2 An Example

Recalling to the same set-up used for the other two procedures (Section 5.3.2), we now provide a numerical example adopting a control pin strategy.
No variable definition is required for this synthesis. The parameter normalization is performed through the `cheb`-type sub-circuit defined above, using a proper indexing as required for a multivariate model case.

Here follows the SPICE netlist, which is self-explaining[8,9].

```
****************************************
**[SIMULATION FILE (Sim.cir)]**
****************************************
* AC Validation netlist (LTSpice)      *
************************
* Port terminations
[...Chosen Simulation Set Up...]

************************
* Parameters setup
************************

** Parameter n. 1
Vparameter ParCtrl_1 0 2.0000000000000000e+00

************************
* External Macromodel
************************

.inc 'PAR_GSKmodel.cir'
xMODcir
+  1 2 0
+ ParCtrl_1
+ PAR_GSKmodel

****************************************
**[GSK-MODEL FILE (Model.cir)]**
****************************************

.INCLUDE PAR_GSKmodel_num.cir
.INCLUDE PAR_GSKmodel_den.cir

************************
*** BEGIN: parameterized macromodel
************************
.subckt PAR_GSKmodel

** Macromodel Ports/CtrlPins Definition **
+  a_1 a_2 ref
+  CtrlPin_1

************************
* NOTE:
```

[8] Parts inserted inside the script as `[Text]` should be substituted with the specified subject, in a LTSpice compatible form.

[9] The VCCS element `ET1` must be instantiated in the same line. Otherwise, the definition line should be broken and a `+` should be added on top of the first element of the new line. See examples of Section 5.5.1.

```
* a_i  --> input node associated to the port i
* ref  --> reference node, common for all input ports
* CtrlPin_l --> parameter control pins (voltage controlled) associated
    to parameter l
* t_l_i --> input node associated to the chebychev term i of the
   parameter l
************************

*************************
** Parameter Normalization sub-circuit definition**
************************
.subckt cheb CtrlPin_1
+ T1_0 T1_1 T1_2 T1_3 T1_4

** Connect Control Pin to ground **
RCtrP1 CtrlPin_1 0 1e9

** MODEL DEFINED in the FOLLOWING PARAMETER RANGE **
* parMax = 1.8000000000000000e+01
* parMin = 2.0000000000000000e+00

************************
** Polynomial Terms Evaluation **
************************
* Parameter normalization realized with first term *

VT0 T1_0 0 1
RT0 T1_0 0 1e9
ET1 T1_1 0 value={-1+2*(v(CtrlPin_1,0)-2.0000000000000000e+00)
   /(1.8000000000000000e+01-2.0000000000000000e+00)}
RT1 T1_1 0 1e9
ET2 T1_2 0 value={2*v(T1_1,0)*v(T1_1,0)-v(T1_0,0)}
RT2 T1_2 0 1e9
ET3 T1_3 0 value={2*v(T1_1,0)*v(T1_2,0)-v(T1_1,0)}
RT3 T1_3 0 1e9
ET4 T1_4 0 value={2*v(T1_1,0)*v(T1_3,0)-v(T1_2,0)}
RT4 T1_4 0 1e9

.ends cheb

* END of Parameter Normalization sub-circuit
*************************

************************
* Parameter normalization sbckt connection to CtrlPin *
************************
XP CtrlPin_1
+ t_1_0 t_1_1 t_1_2 t_1_3 t_1_4
+ cheb
```

```
************************
* External/output ports of the network
************************
[...Chosen Representation...]

************************
* Numerator network (Cn)
************************
[...Chosen Representation...]

XNN_1 c_n_1 c_n_2 0
+ t_1_0 t_1_1 t_1_2 t_1_3 t_1_4 PAR_GSKmodel_num

************************
* Denominator network (Dn)
************************
*Port: 1
F_d_1 0 c_d_1 Vtmp_n_1 1
XDD_1 c_d_1 0
+ t_1_0 t_1_1 t_1_2 t_1_3 PAR_GSKmodel_den

*Port: 2
F_d_2 0 c_d_2 Vtmp_n_2 1
XDD_2 c_d_2 0
+ t_1_0 t_1_1 t_1_2 t_1_3 PAR_GSKmodel_den

.ends * END parameterized macromodel

****************************************
**[SIMULATION FILE (Sim.cir)]**
****************************************
************************
* Frequency Analysis setup
************************
.ac list
[...Frequency Points...]

.end * END simulation
```

5.6 Function Calls

In this part we document all the functions necessary to produce the examples synthesis of this chapter.
In particular we will see the following:

- GSK_Model2Cir_Par, driver to the other functions;
- MakeGSKWrapper_PARtype, realizes a wrapper sub-circuit for the admittance sub-blocks, as described in Sec. 5.3, Sec. 5.4 and Sec. 5.5.

PARtype is a string depending by the parameter call type required.

5.6.1 GSK_Model2Cir Parametric

This function generates a SPICE sub-circuit that realizes a model in parametrized Generalized Sanathanan-Koerner form.
The input model must be a ratio between numerator and denominator series objects, which must be defined through the 'partialfractions' and 'chebychev' bases.
The *Options* fields enables a customization of the SPICE synthesis, as detailed below.
Here follows the function call.

```
function [NumSS,DenSS,om,Href] = GSK_Model2Cir_Par(Model,pathname,name
    ,Options)
```

The inputs are:

- Model is the model in the gsk form;
- pathname is the path where the output files will be located;
- name SPICE sub-circuit name in the output file *'name.cir'* (Do not include the extension in the input string);
- Options is an (optional) input parameter that includes the fields
  - Options.GroundReferences [default = 0]
    determines how the reference nodes for all ports are generated. If it is set to 0, each port in the synthesized equivalent circuit will have a 'private' (floating) reference node. If it is set to 1, all ports will share a common reference node (useful for grounded multiports).
  - Options.ResistorType [default = 1]
    controls the synthesis of resistors in the equivalent circuit. These resistors are not 'true' resistors, but are just dummy components that are used to translate the model equations into a SPICE netlist. Therefore, these resistors might lead to wrong results when employed in a 'noise' analysis. Four different types of synthesis are available, according to the value of Options.ResistorType:
    1. synthesis as standard resistor (default)
    2. synthesis as a resistor with appended keyword 'noise=0' (available only for HSPICE)
    3. synthesis as current-controlled voltage source
    4. synthesis as voltage-controlled current-source

- Options.mustOptimizeCap [default = 1]
  optimization of the capacitance value based on the location of the model poles (for GHz-range models, typical values are 1nF or 1pF). If false, all capacitances are set to 1F.
- Options.debug [default = 0]
  if true, the function computes a validation of model response from individual numerator and denominator response and return the reference model response and angular frequency samples on output. Otherwise no validation is performed and both om and Href are empty.
- Options.parametrized [default = 0]
  (optional) if true, a parametrized synthesis is performed. An automatic check on the model bases sets its value to 1 if the model is parametric, with 'chebychev' basis.
- Options.ParCall [default = 1]
  controls the parameter call type in the wrapper and admittances sub-circuits synthesis. Three different kind of parameter calls are available, according to the value of Options.ParCall:
  1. independent parameter call. The user can select a value to the i-th parameter ϑ_i through the variable `par_i`, where $i = 1, ..., \rho$ is an index which indicates the "actual" parameter, adopting curly brackets invocations in the macromodel sub-circuit call in the simulation circuit. (default)
  2. control pin definition. Parameter values must be provided by the user as voltage drops on specific pins. All the control pins are defined as last pins of the macromodel sub-circuit.
  3. global parameter call. Users must define a (global) variable, called par_i, in the simulation circuit to select a parameter value ϑ_i for the macromodel. No curly bracket invocation is required during the model sub-circuit call. This global variable directly affects the admittance sub-circuits elements definition.

On output, the function returns:

- NumSS, DenSS state-space realizations of numerator and denominator of input Model. If a parametric synthesis is required, then NumSS and DenSS return the state-space realization obtained evaluating the parameterized model at the first parameter value available in the parameter domain.
- om, Href angular and frequency response. If parametric synthesis required, these variables are empty.

5.6.2 makeGSKWrapper Parametric

Constructs the main interface of the parameterized model, consisting in a wrapper that connects the individual numerator and denominator netlists.
Three functions were developed, depending by the parameter call type. In particular, we have:

- MakeGSKWrapper_GlobalPar, for a global parameter call;

- MakeGSKWrapper_IndPar, for a independent parameter call;
- MakeGSKWrapper_CtrlPin, for a control pin definition.

All of them share the same input definitions.

Here follows the function calls

```
function = makeGSKWrapper_GlobalPar(R0,P,N_name,D_name,pathname,
    cktname,CommonGND,parameterized,parValues,NparNum,NparDen)

function = makeGSKWrapper_IndPar(R0,P,N_name,D_name,pathname,cktname,
    CommonGND,parameterized,parValues,NparNum,NparDen)

function = makeGSKWrapper_CtrlPin(R0,P,N_name,D_name,pathname,cktname,
    CommonGND,parameterized,parValues,NparNum,NparDen)
```

The inputs, common for all of them, are:

- R0 is the reference port resistance and allows to select the model representation (**0** admittance case ; **Inf** impedance case ; **0<R0<Inf** scattering case)
- P is the model ports number
- N_name, D_name are the names of numerator and denominator sub-circuits, which are supposed to be available in individual files with same name and extension '*.cir'
- pathname is the path where wrapper netlist will be located
- cktname is the name of main sub-circuit and corresponding netlist file in the form 'name.cir'. (Do not include the extension in the input string)
- CommonGND is the type of interface port referencing scheme. If set to 0, each port in the synthesized circuit will have a 'private' (floating) reference node. If it is set to 1, all ports will share a common reference node.
- parameterized if 'true', sets the parametric environment. In this case the parameter value (imposed as user-defined variable or by control Pin voltage) is normalized, using the parameter range of definition of the model. If necessary, the sub-circuit call is modified, adding parameter initialization or control pin definition.
- parValues is a cell array of vectors containing all the parameter values, necessary for the normalization.
- NparNum, NparDen Chebychev polynomial order of numerator and denominator bases. Necessary to define the number of parameter bases terms.

Chapter 6

SPICE Synthesis of Parametric Components

We discussed in Chapter 5 the strategies available to provide the parameter ϑ to the admittance sub-blocks, which contain the parameter-dependent circuit elements necessary for an equivalent SPICE synthesis. In particular, we provided not only a way to pass the parameter, but we also proposed a procedure to reduce the number of computations in the components of the circuit.
In fact, we suggested a method to perform the partial evaluation of the parameter-dependent basis function, in order to evaluate each term of $\xi_\ell(\vartheta)$ and provide them to the admittance sub-blocks.

In this Chapter we justify the above procedure, focusing on the problem of synthesizing the parameter-dependent basis $\xi_\ell(\vartheta)$. We investigate several strategies to construct the (parameter-dependent) model coefficients of (2.8), looking for the most efficient and flexible approach.

Moreover, we report the effects of this procedure on the parameter-dependent circuit elements choice in a complete admittance sub-circuit synthesis, seeking for the best one (with LTSpice as working environment).

We support the proposed procedures with numerical examples: in the last section of the chapter the functions calls to reproduce these examples are reported.

6.1 Parameter-Dependent Basis Synthesis

Since the objective of this Chapter is the admittance sub-block components synthesis, starting from the PSK model structure of (2.6), we now refer only to the denominator (scalar) sub-model defined as

$$\mathsf{D}(s,\vartheta) = \sum_{n=0}^{\bar{n}} r_n(\vartheta)\,\varphi_n(s), \tag{6.1}$$

where

$$r_n(\vartheta) = \sum_{\ell=1}^{\bar{\ell}} r_{n,\ell}\, \xi_\ell(\vartheta) \tag{6.2}$$

are denominator model coefficients, with $r_{n,\ell} \in \mathbb{R}$.
We denoted with $\bar{n}$ the frequency basis order and with $\bar{\ell}$ the cardinality of the parameter-dependent basis function.
From Section 5.1, we can denote

$$i_D = \mathsf{D}(s,\vartheta)\, v_D = \sum_{n=0}^{\bar{n}} j_{D,n} \tag{6.3}$$

where $j_{D,0} = r_0(\vartheta)\, v_D$ and

$$j_{D,n} = r_n(\vartheta)\, v_{C,n}, \quad \text{with} \quad v_{C,n} = (s - q_n)^{-1}\, v_D \tag{6.4}$$

In the following we will consider only one external parameter, imposing $\rho = 1$ and simplifying the notation accordingly. Nevertheless, the proposed procedure is general and the extension to either multi-port or multivariate (or both) cases is straightforward.
Moreover, we will refer only to a particular class of orthonormal parameter-dependent basis: the Chebychev polynomials (see Sec 2.1.1 and Sec. 5.2). The strategies we are about to suggest are, however, general enough to be extended to other bases. Focusing on the Chebychev polynomial is just a matter of easier comprehension and explanation.

Starting with the above set-up, we can now investigate the three main alternatives to synthesize the parameter-dependent basis functions and consequently the model coefficients $r_n(\vartheta)$.

(a) For each parameter-dependent element, a proper sub-circuit, which receives as input the normalized parameter $\tilde{\vartheta}$, is realized. The real-valued $r_{n,\ell}$ model coefficients can be either provided as input or defined as a number during the sub-circuits synthesis. The Chebychev polynomial elements are re-computed internally each of this component, through the realization of (6.2). We provide a general example in the following netlist.

```
* Residue of the matrix C n.1
.subckt PAR_GSKmodel_denC_1_1 a_1_1 b_1_1
* Generate Chebychev polynomials
.param y10 = 1.0
.param y11 = y1
.param y12 = 2*y1*y11-y10
** Polynomial coefficients specification **
RC_1_1 a_1_1 b_1_1 {(y10*coeff0 +y11*coeff1 +y12*coeff2)}
.ends
```

The parameter-dependent component is defined in the netlist above as a resistor and takes advantage of a global parameter call to receive the normalized parameter value

through the variable `y1` (see Section 5.3). We assumed the three model coefficients `coeff0`,`coeff1` and `coeff2` fixed: they should be substituted with their numerical values in the netlist example.

(b) The parameter-dependent polynomial terms of $\xi_\ell(\vartheta)$ are computed only once in the main PSK-model file (see parameter call options of Chapter 5). They are provided one at the time, with their corresponding model coefficient $r_{n,\ell}$, as input to a component. This element could be either a proper circuit element or a dedicated sub-circuit. Appropriate adder circuits are created in order to realize the expression of (6.2), by 'linking' these components.
We provide an example of three sub-circuits defined to compute the first three parameter-dependent basis terms in the following netlist.

```
* Generate Chebychev term n. 0
.subckt cheb_0 a_0 b_0 c_0 d_0 params: x0=1
GP0 a_0 b_0 c_0 d_0 { coeff*x0 }
.ends

* Generate Chebychev term n. 1
.subckt cheb_1 a_1 b_1 c_1 d_1 params: x1=1
GP1 a_1 b_1 c_1 d_1 { coeff*x1 }
.ends

* Generate Chebychev term n. 1
.subckt cheb_2 a_2 b_2 c_2 d_2 params: x2=1
GP2 a_2 b_2 c_2 d_2 { coeff*x2 }
.ends
```

where `x0`,`x1` and `x2` are the first three values of the parameter-dependent basis and `coeff` is a model coefficient. We can call the second component using the netlist below

```
XP a_1 0 b_1 0 cheb_1 x1={y1} coeff= 10
```

where the model coefficient $r_{n,\ell}$ is provided as input (`coeff`) and the parameter value is received through an independent parameter call, using the variable `y1` (see Section 5.4).
An alternative to the above sub-circuit example is to realize an equivalent VCCS component, which can be directly inserted in the admittance sub-block, as

```
GP1 a_1 0 b_1 0 { coeff*x1 }
```

that allows avoiding the creation of ℓ sub-circuits.
We recall that the above parameter-dependent components, characterized as two VCCS, are synthesized inside the admittance sub-circuit. Their gain is defined as the voltage drop between the `b_1` pin and the 'local' ground `0` and allows providing directly the n-th term contribution of frequency-dependent basis $\varphi_n(s)$, denoted as $v_{C,n}$ (or v_D if $n = 0$) (6.3). Indeed, we can directly realized the product

$$Gain = v_{C,n} \cdot r_{n,\ell}\xi_\ell(\vartheta) = \varphi_n(s) \cdot r_{n,\ell}\xi_\ell(\vartheta) \tag{6.5}$$

inside each of the above-defined components.

(c) The parameter-dependent polynomial terms of $\xi_\ell(\vartheta)$ are computed only once in the main PSK-model file, as in (b), but they are then provided all together as inputs to a proper parameter-dependent component, which realizes $r_n(\vartheta)$ (6.2). This element can be defined either as a dedicated sub-circuit or as a proper circuit component. As in (b), the coefficients can be provided as input or they can be 'fixed' during the synthesis.
In the following we provide an example

```
* Residue of the matrix C n.1
.subckt PAR_GSKmodel_denC_1_1 a_1_1 b_1_1
* Generate Chebychev polynomials
** Polynomial coefficients specification **
RC_1_1 a_1_1 b_1_1 {(y10*coeff0 +y11*coeff1 +y12*coeff2)}
.ends
```

where a sub-circuit is used to define a parameter-dependent resistor. In such case, the parameter-dependent basis terms (`y10`, `y11` and `y12`) are provided through a global parameter call, while the model coefficients are assumed fixed, as in the (a) example, and should be substituted with their numerical values.

In the examples presented for the several strategies we showed the effects of the parameter call choice on the admittance sub-blocks elements instantiation and definition. We will dedicate a specific Section for the Control Pin call when we will analyze the synthesis of parametric VCCS-elements.

Moreover, in order to apply the strategy (a), which requires to recompute each parameter-dependent basis term, and at the same time to allow a control pin call, a massive number of voltage sources must be added in the overall circuit (see Section 5.5.1). The combination of the two procedures is practically impossible to implement due to the very large runtime required in a simulation.

To conclude, recalling to the examples of Chapter 5, we can now observe that in those netlists we always set an environment compatible with the procedures (b) and (c). To extend those examples to the (a) strategy we should move the `Polynomial Term Evaluation`[1] routine inside each parameter-dependent component, exactly as we did in the first two examples above, and change the admittances instantiations consistently.

6.2 Parameter-Dependent Circuit Elements

We now focus on the parameter-dependent components choice for the admittance sub-block synthesis, investigating several strategies to obtain an equivalent realization of (6.3). Moreover, we recall that the model representation (scattering, admittance or impedance) does not affect the synthesis at this level: only the PSK-model interface circuit is changed according to the selected parameter call option.
In the following we will analyse a scalar case only, referring to the denominator admittance sub-block: the (multiport) numerator realization is straightforward.

[1] See netlists of Chapter 5 for more details, e.g. Example 5.4.2.

6.2.1 Resistors

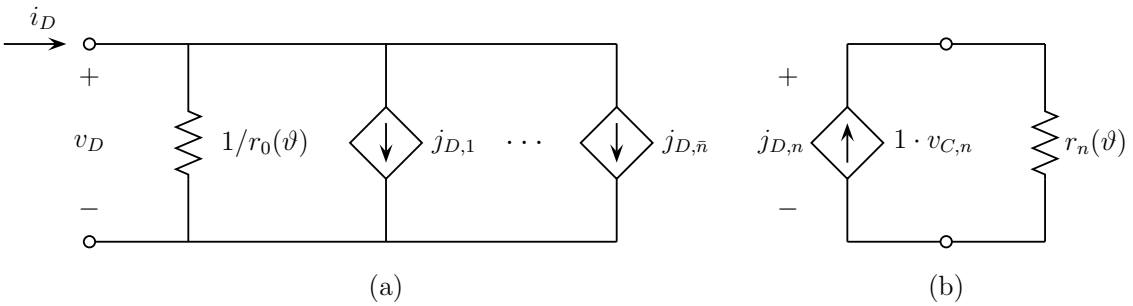


Figure 6.1: Synthesis of a parametrized admittance sub-block for a scalar denominator, using parameter-dependent resistor elements. At least one resistor is realized in the main interface circuit (a). The sub-circuit (b) synthesizes each auxiliary current $j_{D,n}$ and should be repeated n times (with $n = 1, ..., \bar{n}$).

The first and simplest sub-circuit for the admittance sub-block components synthesis is a parametric-dependent resistor.
The alternatives (a) and (c) reported in Section 6.1 can be both applied during the synthesis of this component, as it can be seen in their examples. In any case, we recall that the n-th parameter-dependent resistance corresponds exactly to the model coefficient $r_n(\vartheta)$, necessary to synthesize the auxiliary variable $j_{D,n}$ (6.3).

Nevertheless, the integration of these components into a complete admittance sub-block extraction requires to create some additional sub-circuits with respect to the netlist realized for a non-parametric GSK-model synthesis.
Indeed, the interface circuit of the admittance sub-block can be kept unchanged but each gain of the VCCS elements is computed in a dedicate sub-circuit. This is implemented synthesizing the variable $j_{D,n}$ as the voltage drop on top of the parameter-dependent resistor, when the current value through it is exactly the voltage $v_{C,n}$ of (6.3). This current is realized by using another VCCS element, which closes the additional sub-circuit.
Figure 6.1 shows this kind of realization: we refer to Section 4.1.1 for the model poles synthesis, which is not affected by the parameterization.

A previous version of this synthesis was attempted, avoiding the resistor in the main interface circuit: however, a DC path to ground was not guaranteed in the resulting overall circuit. This is the reason why a parameter-dependent resistor is inserted in Fig 6.1(a): its resistance is defined as in the inverse of the model coefficient $r_0(\vartheta)$, taking advantage of the fact that this variable is related to the constant term $\varphi_0(s)$ of the frequency-dependent partial fraction basis $\varphi_n(s)$ (see Sec. 1.2.1). In a multiport case, this procedure must be repeated for all the diagonal entries of the numerator state space realization matrix $\mathbf{D}_1(\vartheta)$ (see Sec. 2.1.2 for details).

The synthesis of the parameter-dependent components as resistors presents several drawbacks.
First, this technique requires a massive number of additional circuit components with

respect to the non-parametric GSK-model synthesis, which corresponds to a loss of performances during numerical simulations.
Moreover, the strategy (b), reported in Section 6.1, to compute the Chebychev polynomial is not compatible with this sort of element: most of the SPICE engines, in fact, do not admit a null resistance value (in our case, LTSpice does not). Indeed, there is no guarantee that either all the model coefficients or at least the term $r_n(\vartheta)$ will be different from zero. This is a strong assumption for a general synthesis strategy, which may lead to a simulation fail for such particular conditions.
Lastly, this kind of circuit element is not compatible with the use of a control pin: with LTSpice, it is not possible to read a voltage drop from a specific pin in a resistance value definition.

For these reasons, we will investigate the parameter-dependent component synthesis through another kind of circuit element in Section 6.2.2.

An example

We now consider a two port capacitor (**Case 2** of Appendix A) with known scattering responses ($\bar{k} = 191$ frequency samples). This corresponds to the same example reported in Chapter 4 with a main difference: now the capacitor sidelength is no more fixed. A set of eight equally spaced parameter samples ϑ is obtained making it vary between $254\,\mu m$ and $609.6\,\mu m$.

A good validation of the final model vs raw data is obtained using $\bar{n} = 4$ poles for numerator and denominator sub-models, imposing a relaxed normalization. Chebychev polynomial is used as parameter-dependent basis function, imposing the same cardinality both for the numerator and denominator sub-models $\ell_N = \ell_D = \ell = 1$.

We are going to focus our attention on the admittance SPICE synthesis, and in particular, on the parameter-dependent resistor components providing numerical results. We will report a scalar case only, using the denominator model sub-block previously presented: the multiport strategy realization (for the numerator sub-model) is straightforward, starting from Sec.4.2.2, and we will omit it.

For this realization, we use an independent parameter call procedure, evaluating all the Chebychev polynomial terms inside each parameter-dependent sub-circuit: the normalized parameter is provided as input. This procedure corresponds to the (a) strategy of Sec. 6.1.
These sub-circuits are realized in a dedicated file, that is reported in the following[2].

```
************************************
* SPICE subcircuit REALIZATION     *
* This file is automatically generated *
************************************
* Created by cheb2CIR
**********************************
******* Subcircuit definition *****
```

[2] Each line break of an element definition should be corrected, following the netlist rules (adding a + before the first element of a new line)

```
**********************************
** Port : 1
*********************************************************
* ** Subcircuit showing Chebychev polynomial T_n(x) **
*********************************************************
* Residue of the matrix D n.1
.subckt PAR_GSKmodel_denD_1_1 c_1_1 d_1_1 params: x=1
* Generate Chebychev polynomials
.param x0 = 1.0
.param x1 = x
** Polynomial coefficients specification **
RD_1_1 c_1_1 d_1_1 {1/(x0*2.6238563418856629e+00 + x1
    *2.3120136453818994e+00 )}
.ends
*******************
* End of subcircuit
*******************

*********************************************************
* ** Subcircuit showing Chebychev polynomial T_n(x) **
*********************************************************
* Residue of the pole n. 1
.subckt PAR_GSKmodel_denC_1_1 a_1_1 b_1_1 params: x=1
* Generate Chebychev polynomials
.param x0 = 1.0
.param x1 = x
** Polynomial coefficients specification **
RC_1_1 a_1_1 b_1_1 {(x0*1.0580700651906305e-02 + x1
    *-2.4733358979610849e-03 )}
.ends
*******************
* End of subcircuit
*******************

*********************************************************
* ** Subcircuit showing Chebychev polynomial T_n(x) **
*********************************************************
* Residue of the pole n. 2
.subckt PAR_GSKmodel_denC_1_2 a_1_2 b_1_2 params: x=1
* Generate Chebychev polynomials
.param x0 = 1.0
.param x1 = x
** Polynomial coefficients specification **
RC_1_2 a_1_2 b_1_2 {(x0*-3.3818898267224790e-03 + x1
    *-6.2718275245796451e-03 )}
.ends
*******************
* End of subcircuit
*******************
```

```
*********************************************************
* ** Subcircuit showing Chebychev polynomial T_n(x) **
*********************************************************
* Residue of the pole n. 3
.subckt PAR_GSKmodel_denC_1_3 a_1_3 b_1_3 params: x=1
* Generate Chebychev polynomials
.param x0 = 1.0
.param x1 = x
** Polynomial coefficients specification **
RC_1_3 a_1_3 b_1_3 {(x0*7.3335707177552789e-01 + x1*4.4161854263110789
    e-01 )}
.ends
*******************
* End of subcircuit
*******************

*********************************************************
* ** Subcircuit showing Chebychev polynomial T_n(x) **
*********************************************************
* Residue of the pole n. 4
.subckt PAR_GSKmodel_denC_1_4 a_1_4 b_1_4 params: x=1
* Generate Chebychev polynomials
.param x0 = 1.0
.param x1 = x
** Polynomial coefficients specification **
RC_1_4 a_1_4 b_1_4 {(x0*3.7342955666744743e-01 + x1*4.6801569811022586
    e-01 )}
.ends
*******************
* End of subcircuit
*******************
```

These components are then integrated in a complete admittance sub-block synthesis as detailed in Section 6.2.1, including the file previously created. We provide a netlist example in the following.

```
**************************************
* SPICE subcircuit REALIZATION       *
* This file is automatically generated *
**************************************
* Created by SS2Cir__par_R
**************************************
*************************************************
* NOTE:
* a_i  --> input node associated to the port i
* ref  --> reference node, common for all input ports
* y --> parameter value in input to the netlist (normalized)
*************************************************

** Include the chebychev file **
```

```
.INCLUDE PAR_GSKmodel_den_cheb.cir

**********************************
* Interface (ports specification) *
**********************************
.subckt PAR_GSKmodel_den
+  a_1 ref
+  params: x = 1
**********************************
****************************************
* Main circuit connected to output nodes *
****************************************

* Port 1
GC_1_1 a_1 ref NPC_1_1 0 1
GC_1_2 a_1 ref NPC_1_2 0 1
GC_1_3 a_1 ref NPC_1_3 0 1
GC_1_4 a_1 ref NPC_1_4 0 1
XPD_1_1 a_1 ref PAR_GSKmodel_denD_1_1 x={y}
*
****************************************
************************************
* Synthesis of real and complex poles *
************************************

* Real pole n. 1
CS_1 NS_1 0 9.9999999999999998e-13
RS_1 NS_1 0 1.0072196180861735e+03
GS_1_1 0 NS_1 a_1 ref 1.0000000000000000e+00
*
* Real pole n. 2
CS_2 NS_2 0 9.9999999999999998e-13
RS_2 NS_2 0 1.9664657708131386e+01
GS_2_1 0 NS_2 a_1 ref 1.0000000000000000e+00
*
* Complex pair n. 3/4
CS_3 NS_3 0 9.9999999999999998e-13
CS_4 NS_4 0 9.9999999999999998e-13
RS_3 NS_3 0 2.7138694198861572e+01
RS_4 NS_4 0 2.7138694198861572e+01
GL_3 0 NS_3 NS_4 0 3.8108614844969030e-01
GL_4 0 NS_4 NS_3 0 -3.8108614844969030e-01
GS_3_1 0 NS_3 a_1 ref 2.0000000000000000e+00
*
************************************
* Synthesis of the parametric polynomials *
************************************

*C Elements
*Port n. 1
```

```
XPC_1_1 NPC_1_1 0 PAR_GSKmodel_denC_1_1 x={y}
GPC_1 0 NPC_1_1 NS_1 0 1

XPC_1_2 NPC_1_2 0 PAR_GSKmodel_denC_1_2 x={y}
GPC_2 0 NPC_1_2 NS_2 0 1

XPC_1_3 NPC_1_3 0 PAR_GSKmodel_denC_1_3 x={y}
GPC_3 0 NPC_1_3 NS_3 0 1

XPC_1_4 NPC_1_4 0 PAR_GSKmodel_denC_1_4 x={y}
GPC_4 0 NPC_1_4 NS_4 0 1

*D Elements
*Port n. 1
******************************
.ends
********************
* End of subcircuit
********************
```

Since the denominator is scalar, the part of the netlist related to the $\mathbf{D}(\vartheta)$ matrix elements is empty. We recall that the diagonal entries are, in fact, inserted in the main interface circuits of the admittance sub-block: in this case, this procedure corresponds to insert the only resistor in the interface circuit on top of the netlist example.

6.2.2 Controlled Sources

In this section we will expose a different approach to the problem of synthesizing a parameter-dependent component, using Controlled Sources elements. These components, commonly available in any circuit solver, reveal several favourable characteristics for the extraction of a PSK-model, in an LTSpice environment.
In particular, we realize two synthesis procedures based on Voltage Controlled Current Sources (VCCS): the same strategies could be equally realized using Current Controlled Current Sources (CCCS).
Even if this component is compatible with all the alternatives proposed in Section 6.1 for the Chebychev polynomial synthesis, we implement an admittance sub-block extraction following only the last two (denoted as (b) and (c)), due to the expectation of better model performances.
We now report these two different approaches.

(i) Strategy

A major modification of the admittance sub-block interface circuit (with respect to the non-parametrized synthesis) is necessary if each term of the Chebychev polynomial must be provided to a single component ((b) strategy), which in this case is a proper sub-circuit. Indeed, the VCCSs elements must be replaced with CCCSs components. The $j_{D,n}$ auxiliary currents of (6.3) are realized as the sum of the currents obtained from the parallel connection of ℓ parameter-dependent VCCS components, whose gain is defined

as (6.5). This corresponds to an equivalent SPICE realization of the single sum term of (6.3), as it is shown in Figure 6.2.

In the following we provide a netlist example, using the parameter-dependent compo-

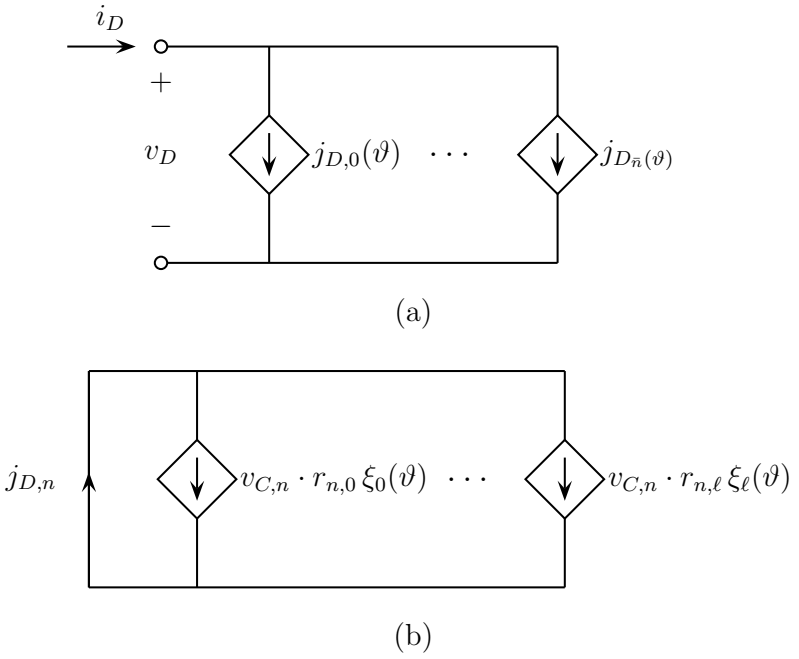


Figure 6.2: Synthesis of a parametrized admittance sub-block for a scalar denominator, using parameter-dependent VCCS elements. Main interface circuit reported in (a). The 'sum' sub-circuit (b) synthesizes each auxiliary current $j_{D,n}$ as the sum of currents provided by the parameter-dependent components. Each one of these elements is related to a parameter-dependent basis term ξ_ℓ.

nents defined in Sec. 6.1(b) , as

```
GPC_1_2_0  NPC_1_2  0  NS_2  0  {y0*[coeff0]}
GPC_1_2_1  NPC_1_2  0  NS_2  0  {y1*[coeff1]}
VPC_1_2  0  NPC_1_2  0
```

where `y0` and `y1` are user defined variables storing the first two chebychev polynomial terms, while `coeff0` and `coeff1` are the model coefficients numerical value.
In the above netlist, the current that flows through the short circuit denoted as `VPC_1_2`, and defined between the node `NPC_1_2` and the 'local' ground (`0`), is equivalent to the expression

$$j_{D,n} = \sum_{\ell=0}^{\bar{\ell}} v_{C,n} \cdot r_{n,\ell} \xi_\ell(\vartheta) = \varphi_n(s) \cdot r_n(\vartheta). \tag{6.6}$$

where $v_{C,n}$ is the voltage drop between the pin `NS_2` and the ground (`0`)[3]. For the above

[3] We refer to the netlist example for the synthesis of a scalar admittance of Section 4.2.1.

example we referred to the second model pole ($n = 2$) and to a parameter-basis cardinality $\bar{\ell} = 1$.

The main drawback of this approach is the massive growth of the model components number, which scales linearly with the definition of the parameter-dependent basis order as $O(\bar{\ell}\bar{n}P^2)$. This drastically affects the equivalent PSK-model performances and lead us to develop the second strategy.

We now provide an example of the first strategy described above.

An Example of (i) Strategy

We now synthesize the same example of Section 6.2.1, adopting a parameter-dependent VCCS component and the above strategy (Sec 6.2.2). In particular, we use a proper dependent source definition, combined with an independent parameter call, to synthesize the admittance sub-block of the model denominator.

We can see the resulting netlist in the following.

```
**************************************
* SPICE subcircuit REALIZATION       *
* This file is automatically generated *
**************************************
* Created by SS2Cir_par_VCCSi
**************************************
************************************************************
* NOTE:
* a_i  --> input node associated to the port i
* ref  --> reference node, common for all input ports
* y --> parameter value in input to the netlist (normalized)
************************************************************
**********************************
* Interface (ports specification) *
**********************************
.subckt PAR_GSKmodel_den
+  a_1 ref
+ y0 = 1 y1 = 1
**********************************
****************************************
* Main circuit connected to output nodes *
****************************************
* Port 1
FC_1_1 a_1 ref VPC_1_1 1
FC_1_2 a_1 ref VPC_1_2 1
FC_1_3 a_1 ref VPC_1_3 1
FC_1_4 a_1 ref VPC_1_4 1
FD_1_1 a_1 ref VPD_1_1 1
*
**************************************
* Synthesis of real and complex poles *
**************************************
* Real pole n. 1
```

```
CS_1 NS_1 0 9.9999999999999998e-13
RS_1 NS_1 0 1.0072196180861735e+03
GS_1_1 0 NS_1 a_1 ref 1.0000000000000000e+00
*
* Real pole n. 2
CS_2 NS_2 0 9.9999999999999998e-13
RS_2 NS_2 0 1.9664657708131386e+01
GS_2_1 0 NS_2 a_1 ref 1.0000000000000000e+00
*
* Complex pair n. 3/4
CS_3 NS_3 0 9.9999999999999998e-13
CS_4 NS_4 0 9.9999999999999998e-13
RS_3 NS_3 0 2.7138694198861572e+01
RS_4 NS_4 0 2.7138694198861572e+01
GL_3 0 NS_3 NS_4 0 3.8108614844969030e-01
GL_4 0 NS_4 NS_3 0 -3.8108614844969030e-01
GS_3_1 0 NS_3 a_1 ref 2.0000000000000000e+00
*
************************************
* Synthesis of the parametric polynomials *
************************************
*C Elements
*Port n. 1
GPC_1_1_0 NPC_1_1 0 NS_1 0 {y0*1.0580700651906305e-02}
GPC_1_1_1 NPC_1_1 0 NS_1 0 {y1*-2.4733358979610849e-03}
VPC_1_1 0 NPC_1_1 0

GPC_1_2_0 NPC_1_2 0 NS_2 0 {y0*-3.3818898267224790e-03}
GPC_1_2_1 NPC_1_2 0 NS_2 0 {y1*-6.2718275245796451e-03}
VPC_1_2 0 NPC_1_2 0

GPC_1_3_0 NPC_1_3 0 NS_3 0 {y0*7.3335707177552789e-01}
GPC_1_3_1 NPC_1_3 0 NS_3 0 {y1*4.4161854263110789e-01}
VPC_1_3 0 NPC_1_3 0

GPC_1_4_0 NPC_1_4 0 NS_4 0 {y0*3.7342955666744743e-01}
GPC_1_4_1 NPC_1_4 0 NS_4 0 {y1*4.6801569811022586e-01}
VPC_1_4 0 NPC_1_4 0

*D Elements
*Port n. 1
GPD_1_1_0 NPD_1_1 0 a_1 ref {y0*2.6238563418856629e+00}
GPD_1_1_1 NPD_1_1 0 a_1 ref {y1*2.3120136453818994e+00}
VPD_1_1 0 NPD_1_1 0

*****************************
.ends
******************
* End of subcircuit
******************
```

(ii) Strategy

A proper circuit element can be defined to receive as inputs all the Chebychev polynomial terms and to internally compute the model coefficient $r_n(\vartheta)$ of (6.2), which corresponds to the strategy (c) of Sec. 6.1. In such case, no modification to the non-parametric admittance sub-block synthesis structure is required: this technique is similar to the one reported in Chapter 5.1 and showed in Fig. 5.1, with an adjustment on the interface circuit.
Indeed, this procedure takes advantage of a new definition of the element gain as

```
Gxxx n+ n- value={<expression>}
```

This is the expression of Arbitrary Behavioral Voltage (or Current) Source (ABVS) in an LTSpice compatible form [1], which can also be defined as `BVxx`, according to the LTSpice manual. With this representation, we are now able to directly realize $j_{D,n}$ of (6.6) in a single component.
This technique can be easily combined with the several parameter call option available (and detailed in Chapter 5). We provide the same example of the first strategy (Sec. 6.2.2), which is now compatible both for an **Independent Parameter** or a **Global Parameter** call, as:

```
GC_1_2 a_1 ref value={v(NS_2,0)*(+y0*[coeff0] +y1*[coeff1])}
```

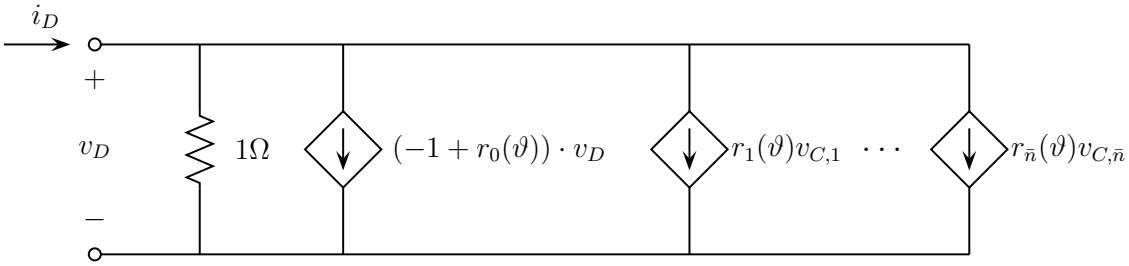
where `y0` and `y1` are defined variables, storing the first two Chebychev polynomial terms values, while `[coeff0]` and `[coeff1]` should be substituted with the model residues numerical values $r_{n,l}$, with $n = 2$ and $\ell = 1{,}2$ respectively. We recall that `a_1` and `ref` are the admittance sub-circuit interface pins.

As we described in Section 4.2, also in this case a resistor must be forced in the main interface circuit in order to avoid a netlist that could lead, in particular conditions, to a circuit configuration that does not present a DC path to ground.
For this reason, we create a unitary 'dummy' resistor and we compensate its effect adding a '-1' term in the trans-conductance definition related to the model coefficient $r_0(\vartheta)$ which, in a multiport case, corresponds to modify each VCCS element associated to a diagonal entry of the model state space matrix $\mathbf{D}(\vartheta)$ (see Sec.2.1.2). The resulting circuit is showed in Figure 6.3
We provide an example of this modification in the netlist reported below.

```
GD_1_1 a_1 ref value={v(a_1,ref)*(-1 +y0*[coeff0] +y1*[coeff1])}
RD_1_1 a_1 ref 1
```

All the assumptions made in the example reported above, about both variables and pins, still hold. This technique presents the main advantage of leaving the number of circuit elements unchanged with respect to a non-parametric GSK-model extraction, with a very slight impact on simulations performances.
Moreover, the same syntax still holds if a CCCS must be defined as the parameter-dependent circuit component, due to the generality of the ABVS component. Nevertheless, we did not develop such synthesis yet.

This second technique, in fact, not only solves the problem of the increasing number of parameter-dependent components but most importantly has allowed us to



(b)

Figure 6.3: Synthesis of a parametrized admittance sub-block of a (scalar) denominator, using parameter-dependent VCCS elements directly inserted in the main interface circuit (a). Unitary resistor forced in the circuit in order to guarantee a DC path to ground.

- extend the automatic extraction to a generic multivariate model in a simple and straightforward way;
- realize a SPICE synthesis adopting a **Control Pin**.

We will detail this two key points in the following sections, providing examples for the above strategy implementation.

Extention to Multivariate Model Synthesis

From now on we will refer only to the second strategy of Sec. 6.2.2, exposing some important improvements that were developed only for this case.
We now focus on the extension of the proposed strategy for the synthesis of a parameter-dependent circuit element in the case of a multivariate PSK-model.
Starting from the VCCS element definition of Sec. 6.2.2(ii), we can modify it using the proper indexing, as described in Chapter 5, in order to internally compute $r_n(\vartheta)$ for a multivariate model representation.

Considering two external parameters ($\rho = 2$), we can extend (6.2) to

$$r_n(\vartheta^{(1)}, \vartheta^{(2)}) = \sum_{\ell_1=1}^{\bar{\ell}_1} \sum_{\ell_2=1}^{\bar{\ell}_2} r_{n,\ell_1,\ell_2}\, \xi_{\ell_1}(\vartheta^{(1)})\, \xi_{\ell_2}(\vartheta^{(2)}) \tag{6.7}$$

where $\bar{\ell}_1$ and $\bar{\ell}_2$ are the cardinalities of the parameter-dependent bases.

We can now modify the definition of the controlled elements reported in Sec. 6.2.2(ii) presenting the same example, extended to two parameters, as

```
GC_1_2 a_1 ref value={v(NS_2,0)*(
      +y10*y20*[coeff00] +y11*y20*[coeff10] +y12*y20*[coeff20]
      +y10*y21*[coeff01] +y11*y21*[coeff11] +y12*y21*[coeff21])}
```

where the variables `y1_l1` and `y2_l2` are associated to the parameter-dependent basis terms of $\xi_{\ell_1}(\vartheta^{(1)})$ and $\xi_{\ell_2}(\vartheta^{(2)})$, respectively.

Imposing $\bar{\ell}_1 = 2\,, \bar{\ell}_2 = 1$ and replacing the elements `coeff_l1_l2` [4] with their corresponding model coefficients r_{n,ℓ_1,ℓ_2} (with $n = 2$), we obtain an equivalent expression of (6.7). We recall that the code fragment above is compatible with both an **independent parameter** or a **global parameter** call.

Its extension for more than two parameters ($\rho > 2$) is straightforward. The above procedure does not modify the overall admittance sub-circuit, which remains the one of Figure 6.3.

An Example of (ii) Strategy

We provide here an example of a multivariate indexing synthesis. Otherwise, we extract the same scalar denominator sub-model used in Sec.6.2.1 and Sec.6.2.2, which is dependent by only one external parameter: this choice allows us to obtain a comparison between all the parameter-dependent synthesis strategies. The extension of this synthesis procedure to a 'proper' multivariate case (with $\rho > 1$) is straightforward and does not require any additional observation to the ones reported above.
We adopt an independent parameter call and we provide each Chebychev polynomial term to the parameter-dependent components of the admittance sub-block (strategy(c)).

Here follows the denominator sub-block netlist.

```
**************************************
* SPICE subcircuit REALIZATION       *
* This file is automatically generated *
**************************************
* Created by SS2Cir_par_IndPar
**************************************
*****************************************************
* NOTE:
* a_i  --> input node associated to the port i
* ref  --> reference node, common for all input ports
* yl --> parameter value in input to the netlist (normalized),
   associated to parameter l
*****************************************************
**********************************
* Interface (ports specification) *
**********************************
.subckt PAR_GSKmodel_IndPar_den
+  a_1 ref
+ y10 = 1 y11 = 1
**********************************
****************************************
* Main circuit connected to output nodes *
****************************************

* Port 1
```

[4] The indexes ℓ_1 and ℓ_2 should substitute the 'netlist indexes' `*_l1` and `*_l2`, respectively.

```
GC_1_1 a_1 ref value={v(NS_1,0)*(
 +y10*1.0580700651906305e-02 +y11*-2.4733358979610849e-03)}

GC_1_2 a_1 ref value={v(NS_2,0)*(
 +y10*-3.3818898267224790e-03 +y11*-6.2718275245796451e-03)}

GC_1_3 a_1 ref value={v(NS_3,0)*(
 +y10*7.3335707177552789e-01 +y11*4.4161854263110789e-01)}

GC_1_4 a_1 ref value={v(NS_4,0)*(
 +y10*3.7342955666744743e-01 +y11*4.6801569811022586e-01)}

RD_1_1 a_1 ref 1
GD_1_1 a_1 ref value={v(a_1,ref)*(
 + -1 +y10*2.6238563418856629e+00 +y11*2.3120136453818994e+00)}
*
****************************************
************************************
* Synthesis of real and complex poles *
************************************

* Real pole n. 1
CS_1 NS_1 0 9.9999999999999998e-13
RS_1 NS_1 0 1.0072196180861735e+03
GS_1_1 0 NS_1 a_1 ref 1.0000000000000000e+00
*
* Real pole n. 2
CS_2 NS_2 0 9.9999999999999998e-13
RS_2 NS_2 0 1.9664657708131386e+01
GS_2_1 0 NS_2 a_1 ref 1.0000000000000000e+00
*
* Complex pair n. 3/4
CS_3 NS_3 0 9.9999999999999998e-13
CS_4 NS_4 0 9.9999999999999998e-13
RS_3 NS_3 0 2.7138694198861572e+01
RS_4 NS_4 0 2.7138694198861572e+01
GL_3 0 NS_3 NS_4 0 3.8108614844969030e-01
GL_4 0 NS_4 NS_3 0 -3.8108614844969030e-01
GS_3_1 0 NS_3 a_1 ref 2.0000000000000000e+00
*
****************************
.ends
******************
* End of subcircuit
******************
```

6.3 Synthesis with the Control Pin Interface

Another important advantage given by the last parameter-dependent element definition of Sec. 6.2.2 is its directly compatibility with a *Control Pin* interface.

Indeed, the overall admittance sub-circuit does not change with respect to the one reported in Figure 6.3. We recall that the only synthesis modification must be performed in the wrapper netlist, adding a sub-circuit to normalize the parameter, to evaluate and to return the parameter-dependent basis components, as detailed in Section 5.5.1.

In the following, we use the same example reported in Section 6.2.2 for the second strategy (ii), with the proper indexing adjustments for a multivariate case (even if we consider only one external parameter).
The resulting netlist is:

```
GC_1_1 a_1 ref value={v(NS_1,0)*(
      +v(y10,0)*[coeff0] +v(y11,0)*[coeff1])}
```

where `y10` and `y11` are actually control input pins of the admittance sub-blocks (see Sec.5.5.1) and `0` is the 'local' ground. The parameter-dependent basis terms $\xi_0(\vartheta)$ and $\xi_1(\vartheta)$, previously evaluated, are provided as voltage drops on top of these pins. We recall that `[coeff0]` and `[coeff1]` are the model coefficients $r_{n,\ell}$ and should be substituted with their numerical values.

The above example is completely equivalent to the one of Sec. 6.2.2(ii), reported in Figure 6.3, and produces the same results in terms of model accuracy. Its extension to a multivariate case is straightforward, following the appropriate indexing rules, which are specified in Section 6.2.2.
For what concerns the performances of the equivalent netlists, the control pin case shows a (slight) loss of speed during simulations. This change can be associated either with the increased number of overall components or to the new elements definitions: further investigations should be performed in this direction, stressing the procedure with other circuit solvers.

6.3.1 Topological Issues

Also using a Control Pin interface, a 'dummy' resistor must be added to the main interface circuit of the admittance sub-block. This is necessary to avoid particular circuit configurations that do not present a DC path to ground.
As for the second strategy of Sec. 6.2.2 we compensate the effect of such resistors adding a '-1' term in the definition of the parameter-dependent VCCS elements. This corresponds to the following code fragment:

```
GD_1_1 a_1 ref value={v(a_1,ref)*(
      +v(y10,0)*[coeff0] +v(y11,0)*[coeff1] -1)}
RD_1_1 a_1 ref 1
```

where all the assumptions made for the example of Sec. 6.2.2(ii) still hold.
But a complete model synthesis with the above procedure generates the following warning in the log file:

```
Singular matrix:  Check node modcir:c_d_1
Iteration No. 1
Direct Newton iteration failed to find .op point.  (Use ".option
    noopiter" to skip.)
Starting Gmin stepping
Gmin = 10
Gmin = 1.07374
Gmin = 0.115292
Gmin = 0.0123794
Gmin = 0.00132923
Gmin = 0.000142725
Gmin = 1.5325e-005
Gmin = 1.6455e-006
Gmin = 1.76685e-007
Gmin = 1.89714e-008
Gmin = 2.03704e-009
Gmin = 2.18725e-010
Gmin = 2.34854e-011
Gmin = 2.52173e-012
Gmin = 2.70769e-013
Gmin = 0
Gmin stepping succeeded in finding the operating point.
```

From what it can be seen, LTSpice detects a singularity in the overall circuit that corrupts and lead to a failure of the standard procedure (Direct Newton Method) used to find the operating point (Op) of the circuit.
Indeed, most of the circuit solvers, if an AC analysis is required, automatically perform a pre-processing DC solution of the circuit under investigation: this procedure presents several numerical advantages which increase the simulation performances of the solver.
LTSpice uses, at first, a successive linear approximation to find the Op of a circuit through standard Newton-Raphson iteration ("Direct Newton Iteration"): if this fails, the solver tries several other methods (see LTSpice manual [1]). In our case, an "Adaptive Gmin Stepping" procedure reaches the goal of finding the operative point.
The failure to find the Op point leads to a significant slow-down of the model performances, increasing the time required for a simulation.

Several attempts were tried in order to bypass the issue reported before, since the overall circuit is not singular. The only apprach that was successful is detailed next.

By removing one of the parameter-dependent basis terms from the definition of the parametric VCCS element associated to the model coefficient $r_0(\vartheta)$, which includes the '-1' term (as discussed above), we discovered that the op point computation succeeded without any error or warning.
Moreover, we verified that this behaviour holds for all the parameter-dependent basis terms randomly chosen (in the same element described above).
Starting from the above observations and using the Chebychev polynomial recurrence

property (5.4), we eliminated from the definition of the VCCS element described above the first parameter-dependent basis terms `y_i0`, which in this particular case are always unitary: we repeated this procedure for all the parameters and for all the diagonal entries of the state space matrix $\mathbf{D}(\vartheta)$. The result is presented, using the same example reported above, in the following code fragment:

```
GD_1_1 a_1 ref value={v(a_1,ref)*(
        +[coeff0] +v(y11,0)*[coeff1] -1)}
RD_1_1 a_1 ref 1
```

A note of attention is placed here.
By integrating this procedure in the overall PSK-model synthesis with a Control Pin interface, we solved the problem related to the Op computation in LTSpice, but we lost the generality of the automatic tool developed, which is now strictly connected to the Chebychev basis.
Moreover, this procedure may lead to possible simulations failure using other solvers: further investigations must be performed.

Nevertheless, this strategy was successful in all the stress-tests used in LTSpice to verify the efficiency of the synthesis with a Control Pin interface.

6.3.2 An Example

We provide now the same example reported for the multivariate strategy of Sec. 6.2.2, adopting a control pin interface. The same assumptions made above still hold.

The example netlist, quite similar to the one presented for the multivariate synthesis, is provided in the following.

```
**************************************
* SPICE subcircuit REALIZATION       *
* This file is automatically generated *
**************************************
* Created by SS2Cir_par_CtrlPin
**************************************
**************************************************
* NOTE:
* a_i  --> input node associated to the port i
* ref  --> reference node, common for all input ports
* yi --> input node associated to the i chebychev polynomial term (
    normalized)
**************************************************
*********************************
* Interface (ports specification) *
*********************************
.subckt PAR_GSKmodel_CtrlPin_den
+  a_1 ref
+ y10 y11
*********************************
****************************************
* Main circuit connected to output nodes *
****************************************
```

```
* Port 1
GC_1_1 a_1 ref value={v(NS_1,0)*(
 +v(y10,0)*1.0580700651906305e-02 +v(y11,0)*-2.4733358979610849e-03)}

GC_1_2 a_1 ref value={v(NS_2,0)*(
 +v(y10,0)*-3.3818898267224790e-03 +v(y11,0)*-6.2718275245796451e-03)}

GC_1_3 a_1 ref value={v(NS_3,0)*(
 +v(y10,0)*7.3335707177552789e-01 +v(y11,0)*4.4161854263110789e-01)}

GC_1_4 a_1 ref value={v(NS_4,0)*(
 +v(y10,0)*3.7342955666744743e-01 +v(y11,0)*4.6801569811022586e-01)}

GD_1_1 a_1 ref value={v(a_1,ref)*(
 +2.6238563418856629e+00 +v(y11,0)*2.3120136453818994e+00 -1 )}
RD_1_1 a_1 ref 1
*
****************************************
************************************
* Synthesis of real and complex poles *
************************************

* Real pole n. 1
CS_1 NS_1 0 9.9999999999999998e-13
RS_1 NS_1 0 1.0072196180861735e+03
GS_1_1 0 NS_1 a_1 ref 1.0000000000000000e+00
*
* Real pole n. 2
CS_2 NS_2 0 9.9999999999999998e-13
RS_2 NS_2 0 1.9664657708131386e+01
GS_2_1 0 NS_2 a_1 ref 1.0000000000000000e+00
*
* Complex pair n. 3/4
CS_3 NS_3 0 9.9999999999999998e-13
CS_4 NS_4 0 9.9999999999999998e-13
RS_3 NS_3 0 2.7138694198861572e+01
RS_4 NS_4 0 2.7138694198861572e+01
GL_3 0 NS_3 NS_4 0 3.8108614844969030e-01
GL_4 0 NS_4 NS_3 0 -3.8108614844969030e-01
GS_3_1 0 NS_3 a_1 ref 2.0000000000000000e+00
*
****************************
.ends
******************
* End of subcircuit
******************
```

6.4 Function Calls

In this part, we document all the functions necessary to produce the examples of this chapter. Moreover, we also provide the functions that are implemented to realize the combinations of strategies, which are not supported with netlist examples but are the objectives of the investigations above. We will use all of them in Chapter 7 to compare the final results.
In particular, we will follow the chapter organization, dividing the functions presentation according to the parameter-dependent component used.
We will see the following:

1. Resistor synthesis, which can be realized for the (a) and (b) strategies, with an independent parameter call, as in the example of Sec. 6.2.1;

2. VCCS synthesis, that can be realized for the (a) strategy and for the procedure (b), as a sub-circuit component or as a proper circuit element (see Sec.6.2.2);

3. Multivariate synthesis, which is the final version of the above functions and imply a VCCS component. This can be realized for all the parameter call procedures, including the Control Pin interface.

While the last set of functions can be used with the drivers documented in Chapter 5, the first two sets 'live' in their own environment: proper driver functions allow realizing a complete model synthesis. We will avoid to detail also these drivers since their call is just a matter of nomenclature.

6.4.1 Resistor Synthesis Functions

We now analyse the functions necessary to synthesize an admittance sub-block using a parameter-dependent resistor.
In particular we will see the following:

- cheb2CIR_RStrategyPARType, realizes the parameter-dependent components sub-circuits, according to both the parameter call and the synthesis of the designated $\xi_\ell(\vartheta)$ basis;

- SS2Cir_par_RStrategyPARType, realizes a parameter-dependent sparse synthesis according to the function above.

Strategy is a string depending on the procedure used to synthesize the parameter-dependent polynomial. Recalling to Sec. 6.1, we can summarized them as:

- a , the normalized parameter is provided as an input of a dedicated sub-circuit, which evaluates each parameter-dependent basis term and recomputes the model coefficient $r_n(\vartheta)$;
- b , a proper sub-circuit is defined in order to get as input only one parameter-dependent basis term;
- c , each parameter-dependent basis term is provided as an input of either a dedicated sub-circuit or a proper circuit component, which computes the model coefficient $r_n(\vartheta)$.

PARtype is a string depending on the parameter call type required, and in particular could be set to

- GP, if a compatible realization to the global parameter call is created (see Sec. 5.3);
- IP, if an independent parameter call synthesis is realized (see Sec. 5.4);

cheb2CIR

This class of functions defines the parameter-dependent sub-circuits, which compute the model coefficient $r_0(\vartheta)$, in a dedicated additional file. The element instantiation is realized according to the chosen parameter call and to the selected synthesis for the parameter-dependent basis.
The input model must be a ratio between numerator and denominator series objects, which must be defined through the 'partialfractions' and 'chebychev' bases: the denominator must be a single factor equal to 1.
The *Options* fields enables a customization of the SPICE synthesis, as detailed below.
Here follows the functions calls.

```
function cheb2CIR_RaIP(Model,pathname,name,Options,Cap)
function cheb2CIR_RcIP(Model,pathname,name,Options,Cap)
```

All of them share the same input definitions.

The inputs are:

- Model is the model in the psk form;
- pathname is the path where the output files will be located;
- name SPICE sub-circuit name in the output file *'name_cheb.cir'*;
- Options is used to realized a sub-circuit consistent with the main admittance sub-block and includes the following fields
  - Options.Normalized [default = 1]
    if it is set to 1, the model coefficients are normalized using the normconstant value from the numerator basis;
  - Options.mustOptimizeCap [default = 1]
    if set to 1 enables the optimization of the model coefficients based on the capacitance value *Cap*, that is received as input and is used during the admittance sub-circuit synthesis. If false, *Cap* is set to 1.
  - Options.debug [default = 1]
    if set to 1, enables the realization of a debug file, that is saved as 'test_name_cheb' in the same directory specified by pathname .
- Cap capacitance value, necessary to optimize the model coefficients. If *Options.mustOptimizeCap = 1* and this variable is not provided as input, an error is generated. [default = 1]

SS2Cir_par

This function enables to synthesize a parameter-dependent SPICE netlist of a sub-block, starting from the non-parametric state space realization of a PSK-model, which is evaluated in the first parameter value available. The parameter-dependent components are

added to the synthesis as external sub-circuits, which are generated by a specific function. All the sub-blocks are synthesized following an independent parameter call strategy.
The only dynamic elements included in the synthesis are identical capacitances. The capacitance value can be provided via field *Cap* of *MOD*. If this field is not present, a unitary value is used.

Here follows the function call

SS2Cir_par_RaIP (MOD, pathname , name , Options)
SS2Cir_par_RcIP (MOD, pathname , name , Options , Npar)

All of them share the same input definitions.

The inputs are:

- MOD variable that defines the state-space equations by fields MOD.A, MOD.B, MOD.C and MOD.D. The number of ports can be deduced by rows of C,D or columns of B,D. The field MOD.R0 is used to distinguish between admittance (MOD.R0 == 0) and scattering representation. In this second case the reference resistance used for all ports must be the same and it is also stored in MOD.R0. The capacitance value can be provided via field MOD.Cap. If this is not present, a unitary valued is used.
- pathname the path where the output files will be located
- name SPICE sub-circuit name in the output file *'name.cir'* (Do not include the extension in the input string)
- Options is an (optional) input parameter that includes the fields
  - Options.GroundReferences [default = 0]
    is an additional option that determines how the reference nodes for all ports are generated. If it is set to 0, each port in the synthesized equivalent circuit will have a 'private' (floating) reference node. If it is set to 1, all ports will share a common reference node (useful for grounded multiports).
  - Options.ResistorType [default = 1]
    controls the synthesis of resistors in the equivalent circuit. These resistors are not 'true' resistors, but are just dummy components that are used to translate the model equations into a SPICE netlist. Therefore, these resistors might lead to wrong results when employed in a 'noise' analysis. Four different types of synthesis are available, according to the value of Options.ResistorType:
    1. synthesis as standard resistor (default)
    2. synthesis as a resistor with appended keyword 'noise=0' (available only for HSPICE)
    3. synthesis as current-controlled voltage source
    4. synthesis as voltage-controlled current-source
  - Options.parameterized [default = 0]
    if set to 1, enables a parametric SPICE equivalent synthesis. Otherwise, a non-parametric sparse synthesis of the sub-block is realized.
- Npar (only for (c) strategy)
  is the cardinality of parameter-dependent basis function ℓ. It is necessary to provide the parameter-dependent basis terms as inputs to the parameter-dependent sub-circuits.

6.4.2 VCCS Synthesis Functions

We now analyse the functions necessary to synthesize an admittance sub-block using a Voltage Controlled Voltage Source (VCCS) for each parameter-dependent basis term. This set of functions enables to reproduce the (i) strategy of Section 6.2.2.
In particular we will see the following:

- chebSPICE_VCCSStrategyPARType, realizes the parameter-dependent components sub-circuits, according to both the parameter call and the synthesis of the designated $\xi_\ell(\vartheta)$ basis; moreover, it provides the reshaped model coefficients as output;

- SS2Cir_par_VCCSStrategyPARType, realizes a parameter-dependent sparse synthesis.

where Strategy and PARType are the same strings defined above for the parameter-dependent resistor synthesis.

chebSPICE

This set of functions could generate a dedicated SPICE sub-circuit for each Chebychev polynomial term, which is stored in an additional file. In this case, a single model coefficient is an input argument of the VCCS sub-circuit: it must be provided through the input variable `coeff` (see Sec. 6.2.2 (i)).
The number of Chebychev terms that is used for the synthesis is the maximum between the numerator and denominator parameter-dependent bases.
The input model must be a ratio between numerator and denominator series objects, which must be defined through the 'partialfractions' and 'chebychev' bases.
This function reshapes the model coefficients and provides them as output.

Here follows the function call:

```
function [chebNumCoeff,chebDenCoeff] = chebSPICE_VCCSaIP(Model,
    pathname,name,Options,Cap)
function [chebNumCoeff,chebDenCoeff] = chebSPICE_VCCSbIP(Model,
    pathname,name,Options,Cap,mustExportSPICE)
```

All of them share the same input definitions.
The inputs are:

- Model is the model in the psk form;
- pathname is the path where the output files will be located;
- name SPICE sub-circuit name in the output file *'name_cheb.cir'*;
- Options is used to realized a sub-circuit consistent with the main admittance sub-block and includes the following fields

  - Options.Normalized [default = 1]

    if it is set to 1, the model coefficients are normalized using the normconstant value from the numerator basis;

  - Options.mustOptimizeCap [default = 1]
    if set to 1 enables the optimization of the model coefficients based on the capacitance value *Cap*, provided as input and used during the admittance sub-circuit synthesis. If false, *Cap* is set to 1.
  - Options.debug [default = 1]
    if set to 1, enables the realization of a debug file, that is saved as 'test_name_cheb' in the same directory specified by pathname .

- Cap is the capacitance value, necessary to optimize the model coefficients. If *Options.mustOptimizeCap = 1* and this variable is not provided as input, an error is generated. [default = 1]
- mustExportSPICE (only for (b) strategy)
  if set to 1 enables the generation of an output file that comprehends the instantiations of the VCCS parameter-dependent sub-circuits. [default = 0]

On output, the function returns:

- chebNumCoeff, chebDenCoeff
  are the numerator and denominator model coefficients reshaped in a matrix $A_{n,\ell} \in \mathbb{R}^{\bar{n}\times\bar{\ell}}$: each row is related to a frequency's basis term φ_n while each column is related to a Chebychev polynomial term ξ_ℓ.

SS2Cir_par

This function enables to synthesize a parameter-dependent SPICE netlist of a sub-block, starting from the non-parametric state space realization of a PSK-model, which is evaluated in the first parameter value available.
The parameter-dependent components are added to the synthesis either as external sub-circuits (which are generated by the *chebSPICE* function) or as proper VCCS elements, that are defined as in Sec 6.2.2. All the sub-blocks are synthesized following an independent parameter call strategy.
The only dynamic elements included in the synthesis are identical capacitances. The capacitance value can be provided via field *Cap* of *MOD*. If this field is not present, a unitary value is used.

Here follows the functions calls.

```
function SS2Cir_par_VCCSaIP(MOD, chebCoeff, pathname, name, Options)
function SS2Cir_par_VCCSbIP(MOD, chebCoeff, pathname, name, Options)
```

All of them share the same input definitions.

The inputs are the same of the similar function defined (above) for the synthesis with a parameter-dependent resistor (see Sec. 6.4.1), with the exception of the following input

- chebCoeff stores the (scalar) sub-model coefficients reshaped in a matrix $A_{n,\ell} \in \mathbb{R}^{\bar{n}\times\bar{\ell}}$, which become a tensor for multiport sub-block case; considering a scalar example, each row is related to a frequency's basis term φ_n while each column is related to a Chebychev polynomial term ξ_ℓ;

6.4.3 Multivariate Case Synthesis Functions

We now analyse the functions necessary to synthesize an admittance sub-block using a Voltage Controlled Voltage Source (VCCS) as parameter-dependent component for a multivariate case. This set of functions enables to reproduce the second strategy of Sec. 6.2.2 for a multivariate case (Section 6.2.2) and it is compatible with all the parameter call strategies available (see Chapter 5).
In particular we will see in the following:

- chebSPICE_MultiPar_Tables, reshapes the model coefficients and provides a mapping table of them;
- SS2Cir_par_PARType_Multi, realizes a parameter-dependent sparse synthesis according to parameter call strategy selected.

where PARType is a string depending on the parameter call type required.

chebSPICE_MultiPar_Tables

This function reshapes the model coefficients and provides a mapping table of them. This is necessary to synthesize a multivariate model with a number of external parameters that are not known 'a priori' in the most efficient way: the mapping table, in fact, allows to reduce the number of operations of an entire equivalent SPICE synthesis. We provide an example of the model coefficients matrix, which in the (multiport) numerator case is a tensor, for the (scalar) denominator in Table 6.1: the corresponding mapping table is reported in Table 6.2.

An optimization of the model coefficients using the input capacitance value *Cap* is perfomed (if required). A normalization of the coefficients is realized using the 'normconst' value extracted from the parameter bases.

Here follows the function call

```
function [chebNumCoeff, chebDenCoeff] = chebSPICE_MultiPar_Tables(Model
    , pathname, name, Options, Cap, mustExportSPICE)
```

The inputs are:

- Model is the model in the psk form;
- pathname is the path where the output files will be located;
- name SPICE sub-circuit name in the output file *'name_cheb.cir'*;
- Options is used to realized a sub-circuit consistent with the main admittance sub-block and includes the following fields
  - Options.Normalized [default = 1]
    if it is set to 1, the model coefficients are normalized using the normconstant value from numerator basis;
  - Options.mustOptimizeCap [default = 1]
    if set to 1 enables the optimization of the model coefficients based on the capacitance value *Cap*, provided as input and used during the admittance sub-circuit synthesis. If false, *Cap* is set to 1.

  - Options.debug [default = 1]
    if set to 1, enables the realization of a debug file, that is saved as 'test_name_cheb' in the same directory specified by pathname .

- Cap is the capacitance value, necessary to optimize the model coefficients. If *Options.mustOptimizeCap = 1* and this variable is not provided as input, an error is generated. [default = 1]
- mustExportSPICE NO MORE USED
  if set to 1 enables the generation of an output file, which comprehends the instantiations of the VCCS parameter-dependent sub-circuits. [default = 0]

On output, the function returns:

- chebNumCoeff, chebDenCoeff are two cell arrays of three elements which contain

  1. the model coefficients reshaped in a matrix $A_{n,\ell} \in \mathbb{R}^{\bar{n}\times\bar{\ell}}$, denoting with $\bar{\ell}$ the product of the cardinality of all the parameter-dependent basis and with $\bar{n}$ the number of model poles. Each matrix entries is related to a model coefficient $r_{n,\ell}$. We denote as $\ell = 1, ..., \bar{\ell}$ the column index, while the rows are related to frequency's basis terms $\varphi_n(s)$;
  2. the matrix $B_{i,\ell} \in \mathbb{R}^{\rho\times\bar{\ell}}$, where $i = 1, ..., \rho$ (ρ number of external parameters) and $\ell = 1, ..., \bar{\ell}$. This matrix corresponds to the mapping table of the Chebychev basis function coefficients for each external parameters. The ℓ-th column stores the parameter-dependent basis term order (of the i-th parameter) corresponding to the coefficient $r_{n,\ell}$, which is stored in the matrix of the first output array component.
  3. the vector $c = (\ell_1, ..., \ell_\rho)$, which stores the Chebychev basis functions cardinalities of the external parameters.

	z_0						z_1					
	y_0			y_1			y_0			y_1		
	x_0	x_1	x_2	x_0	x_1	x_2	x_0	x_1	x_2	x_0	x_1	x_2
q_0	$r_{0,1}$	$r_{0,2}$	$r_{0,3}$	$r_{0,4}$	$r_{0,5}$	$r_{0,6}$	$r_{0,7}$	$r_{0,8}$	$r_{0,9}$	$r_{0,10}$	$r_{0,11}$	$r_{0,12}$
q_1	$r_{1,1}$	$r_{1,2}$	$r_{1,3}$	$r_{1,4}$	$r_{1,5}$	$r_{1,6}$	$r_{1,7}$	$r_{1,8}$	$r_{1,9}$	$r_{1,10}$	$r_{1,11}$	$r_{1,12}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
q_9	$r_{9,1}$	$r_{9,2}$	$r_{9,3}$	$r_{9,4}$	$r_{9,5}$	$r_{9,6}$	$r_{9,7}$	$r_{9,8}$	$r_{9,9}$	$r_{9,10}$	$r_{9,11}$	$r_{9,12}$

Table 6.1: First element of the output array *chebDenCoeff*, which is the above matrix denoted as $A_{n,\ell} \in \mathbb{R}^{\bar{n}\times\bar{\ell}}$. We indicate with $r_{n,\ell}$ the (denominator) model coefficients. In the example above, we define three external parameters, denoted as x, y and z respectively. The overall parameter-dependent cardinality is $\bar{\ell} = 3 \cdot 2 \cdot 2 = 12$. We indicate with q_n the model poles, where $n = 1, ..., \bar{n}$ and $\bar{n} = 9$. Each matrix row corresponds to a frequency-dependent basis term, while the columns depend by the external parameters. The extension to the (multiport) numerator case is a tensor.

	z_0						z_1					
	y_0			y_1			y_0			y_1		
	x_0	x_1	x_2	x_0	x_1	x_2	x_0	x_1	x_2	x_0	x_1	x_2
x	0	1	2	0	1	2	0	1	2	0	1	2
y	0	0	0	1	1	1	0	0	0	1	1	1
z	0	0	0	0	0	0	1	1	1	1	1	1

Table 6.2: Second element of the output array *chebDenCoeff*, which is the above mapping table denoted as $B_{i,\ell} \in \mathbb{R}^{\rho \times \bar{\ell}}$, where $i = 1, ..., \rho$ (ρ number of external parameters) and $\ell = 1, ..., \bar{\ell}$, with $\bar{\ell} = 3 \cdot 2 \cdot 2 = 12$. For this example, we define three external parameters, denoted as x, y and z respectively. Each matrix row corresponds to a different parameter and associates its term with the corresponding model coefficient of Table 6.1. To define each parameter-dependent component during the SPICE synthesis, a loop on the rows of this matrix is sufficient to provide the right i-th parameter term with a minimum number of operations. The equivalent result for the (multiport) numerator is still a matrix.

SS2Cir_Par

This function enables to synthesize a parameter-dependent SPICE netlist of a sub-block, starting from the non-parametric state space realization of a PSK-model, which is evaluated in the first parameter value available.
The parameter-dependent components are proper VCCS elements, that are defined as in Sec 6.2.2 (or Sec. 6.3 for the control pin interface). All the strategies to receive the parameter detailed in Chapter 4 are available.
The only dynamic elements included in the synthesis are identical capacitances. The capacitance value can be provided via field *Cap* of *MOD*. If this field is not present, a unitary value is used.

Here follows the functions calls

```
function SS2Cir_par_IndPar_Multi(MOD, pathname, name, Options, chebCoeff)
function SS2Cir_par_CtrlPin_Multi(MOD, pathname, name, Options, chebCoeff)
function SS2Cir_par_GlobalPar(MOD, pathname, name, Options, chebCoeff)
```

All of them share the same input definitions.

The inputs are:

- MOD variable that defines the state-space equations by fields MOD.A, MOD.B, MOD.C and MOD.D. The number of ports can be deduced by rows of C,D or columns of B,D. The field MOD.R0 is used to distinguish between admittance (MOD.R0 == 0) and scattering representation. In this second case the reference resistance used for all ports must be the same and it is also stored in MOD.R0. The capacitance value can be provided via field MOD.Cap. If this is not present, a unitary valued is used.
- pathname the path where the output files will be located
- name SPICE sub-circuit name in the output file *'name.cir'* (Do not include the extension in the input string)
- Options is an (optional) input parameter that includes the fields

- Options.GroundReferences [default = 0]
  is an additional option that determines how the reference nodes for all ports are generated. If it is set to 0, each port in the synthesized equivalent circuit will have a 'private' (floating) reference node. If it is set to 1, all ports will share a common reference node (useful for grounded multiports).
- Options.ResistorType [default = 1]
  controls the synthesis of resistors in the equivalent circuit. These resistors are not 'true' resistors, but are just dummy components that are used to translate the model equations into a SPICE netlist. Therefore, these resistors might lead to wrong results when employed in a 'noise' analysis. Four different types of synthesis are available, according to the value of Options.ResistorType:
  1. synthesis as standard resistor (default)
  2. synthesis as a resistor with appended keyword 'noise=0' (available only for HSPICE)
  3. synthesis as current-controlled voltage source
  4. synthesis as voltage-controlled current-source
- Options.parameterized [default = 0]
  if set to 1, enables a parametric SPICE equivalent synthesis. Otherwise, a non-parametric sparse synthesis of the sub-block is realized.

- chebCoeff cell array (output of the function *chebSPICE_MultiPar_Tables*) which stores:
  1. the matrix (tensor) related to the reshaped model coefficients;
  2. the parameter-dependent basis terms mapping matrix;
  3. the vector of parameter-dependent basis cardinalities.

Chapter 7

SPICE Results and Bias-Dependent Components

This Chapter is intended to both summarize the previous results concerning a parameterized model SPICE synthesis, and to apply the achieved methodology for active analog blocks of widespread interest in modern mobile SoC architectures design, namely bias-dependent components, provided by an industrial partner (Intel).

Indeed, by documenting the numerical results of the circuit simulation on selected stress cases, we provide in the following a comparison between all the strategies available, which are detailed in Chapter 5 and Chapter 6. This enables us to select a synthesis method aware of the proper characteristics of each one of them, by pointing out the best available choice both for the free parameter call and the parameter-dependent components realization.

The second part of this Chapter is dedicated to the practical application of the stability enforcement method proposed in Chapter 3. The mentioned strategy will be applied, in particular, to non-linear circuit blocks, whose complexity may impair systematic signal and power integrity co-simulations at the system level. A parameterization of these elements with respect to their bias-conditions enables to realize small-signal linearized behavioural macromodels: by combining these results with the above parametric SPICE extraction procedure, powerful applications can be achieved in terms of circuit simulations.

7.1 AC Validation Circuits

First, we are going to provide the AC simulation circuit set-up that we used to validate the automatically generated netlists with respect to the raw data. In a parameterized SPICE extraction, this environment should change according to the parameter call strategy selected: for details on the modifications required in the three cases we refer to the examples of Chapter 5.

We now focus on the main aspects that the two validation circuits have in common, both for a parameter-dependent macromodel and a standard GSK-model. In order to validate the response of a single port, each case requires creating a new file (one for every single

parameter value): the resulting netlist depends on the model representation. Indeed, all the model external ports are:

- liked to a short circuit, in the admittance case;
- left open, in the impedance case;
- connected to a reference resistor R_0, in the scattering case;
- if not excited, connected to ground except the impedance representation that is still left open.

Moreover, the AC source connected to the pin under test must be compliant with the above statements, and in particular:

- is a unitary voltage source, in the admittance case;
- coincides with a grounded current source in the impedance case;
- corresponds to a grounded voltage source with a value defined as $2\sqrt{R_0}$, where R_0 is the reference port resistance used for the scattering representation.

The model equivalent circuit is then included in the simulation, whit the appropriate modifications for the parameter call in the case of a netlist generated from a parameterized macromodel. The list of frequency points from the raw data set $\breve{\mathbf{H}}(\mathrm{j}2\pi f_k, \theta_m)$ is inserted at the end of the simulation, just above the specification of the variables that must be probed, which are chosen depending on the model representation.
We now report an example of the parameter-invariant circuit used to validate the (scattering) GSK-model extracted in Chapter 4[1].

```
****************************************
* AC Validation netlist (LTSpice)      *
* This file is automatically generated *
****************************************
* Created by LTSpiceModel2DUM
****************************************
* Port terminations
****************************************
RI_1 NS NR_1 5.0000000000000000e+01
VI_1 NR_1 1 0
RI_2 0  NR_2 5.0000000000000000e+01
VI_2 NR_2 2 0
****************************************
* AC Source
****************************************
Vsource NS 0 AC 1.4142135623730951e+01
****************************************
* External Macromodel
****************************************
.inc "GSKmodel.cir"
xMODcir
+  1 2 0
+ GSKmodel
****************************************
```

[1] `[List of frequency points]` should be substituted with the numerical values from the raw data set $\breve{\mathbf{H}}(\mathrm{j}2\pi f_k, \theta_m)$.

```
* Analysis setup
**************************************
.ac list
+ 4.9999999999999994e+08
* [ List of frequency points ]
+ 1.0000000000000000e+10
.probe
+  V(1) I(VI_1)
+  V(2) I(VI_2)
.end
```

We now report the simulation circuit necessary to validate the parameter-dependent equivalent SPICE circuit that was generated from the PSK-model used as example in Chapter 6. The resulting netlist is not affected by the parameter-dependent component chosen for the synthesis, but only by the parameter call option selected. In the following, we provide a netlist example by using Independent Parameter Call.

```
**************************************
* AC Validation netlist (LTSpice)      *
* This file is automatically generated *
**************************************
* Created by LTSpiceModel2DUM_Par
**************************************
**************************************
* Port terminations
**************************************
RI_1 NS NR_1 5.0000000000000000e+01
VI_1 NR_1 1 0
RI_2 0  NR_2 5.0000000000000000e+01
VI_2 NR_2 2 0
**************************************
* AC Source
**************************************
Vsource NS 0 AC 1.4142135623730951e+01
**************************************
* External Macromodel
**************************************
.inc "PAR_GSKmodel_IndPar.cir"
xMODcir
+  1 2 0
+ PAR_GSKmodel_IndPar
+ par1 = {parameter_1}
**************************************
* Analysis setup
**************************************
** Parameter n.1 : value n.1
.param parameter_1 = 6.0960000000000002e+02
.ac list
+ 4.9999999999999994e+08
* [ List of frequency points ]
+ 1.0000000000000000e+10
```

```
.probe
+  V(1)  I(VI_1)
+  V(2)  I(VI_2)
.end
```

By imposing `parameter_1 = 6.0960000000000002e+02` we actually realized the same validation netlist of the previous example, which was created starting from a GSK-model.

A note of attention is placed here. In order to compare the elapsed time of simulation for a GSK-model equivalent netlist and the parameter-dependent one generated from a PSK-model, we created a specific netlist for all the ports and for each parameter value. This operation can be avoided by making the parameter value varying, according to the LTSpice manual [1], as

```
.step parameter_1 parMin1 parMax1 nSteps
```

or using a list as

```
.step param parameter_1 list [list of value]
```

where `parMin1` and `parMax1` should be substituted with their numerical values, corresponding to $[\vartheta^i_{min}, \vartheta^i_{max}]$ with $i = 1$, as well as `nSteps`, which is the number of steps required in the parameter sweep. Nevertheless, the results obtained from this sort of simulations were no more compatible with a standard GSK-model validation and the comparison of next section could not be realizable.

7.2 SPICE Extractions Comparison

We now provide the numerical results of the test cases that were used to stress the proposed procedure for a parametric dependent SPICE extraction. We refer to Appendix A for the physical structures under investigation and for the model characteristics, such as the frequency and parameter bases orders, and the maximum errors for all the parameter values and ports responses evaluated in MATLAB.

In order to show a comprehensive comparison of the numerical results, Table 7.1 and Table 7.2 are presented in the following. We will analyse both of them starting from the one related to the parameter call, and by proceeding to the results of synthesis which are characterised by the parameter-dependent component: we will provide thus a complete overview of the automatic synthesis extraction results.

7.2.1 Parameter Call Results

We now investigate the results of the parameter call strategies of Chapter 5, which are all implemented with a parameter-dependent VCCS. Table 7.1 provides a criterion for comparison between all the three procedures available to provide the parameter. The results below present some test cases with a general increasing complexity trend both in terms of model bases orders and the number of ports.
This is reflected on the circuit matrix dimension, which is actually not an accurate indicator of the circuital model speed: the overall simulation time is, besides, affected by the strategy chosen to manage the parameter. As expected, a general slow-down of the model

performances with the growth of the parameter-dependent bases order can be noticed.
The above two assumptions are verified by the numerical results: the first column of Table 7.1 provides a parameter-free SPICE realization, which is actually slightly faster than a parameterized netlist.
Nevertheless, it is proved in the following that this slow-down is actually not so relevant. By interpreting the data, in fact, it must be considered that:

- a roundoff error usually occurs in 'small' test cases for a parameter-free netlist, due to their high speed, such that the 'elapsed time' field in the log file (generated by SPICE) presents a null value; this condition does not happen with the same frequency in the 'big' examples, where the time gap between the synthesis is always under the 50% of the overall elapsed time for the non-parameterized netlist; the control pin represents an exception that we will investigate further in the following;

- a slight speed-up of the parameterized macromodel can be provided realizing the modification of the parameter instantiation for the simulation netlist discussed at the end of Section 7.1; we recall that this procedure was not implemented here in order to provide a common criterion for comparison of the presented cases.

Moreover, by analysing the results it can be noticed that the accuracy of the extracted models does not change with respect to the non-parameterized equivalent realizations.
This statement is supported through the maximum relative error e_{rel}, which is a basis for comparison between the circuit simulation and the MATLAB results: this value always presents the same order of magnitude both for the parameterized and parameter-free netlists, which demonstrates that the proposed SPICE synthesis procedure does not affect the overall accuracy.

To conclude, the control pin case presents, as expected, a growth of the time required for simulations and a slight increase of the matrix dimension. This is related to the normalization circuit (see Sec. 5.5.1 for details), which must insert some additional components providing the parameter-dependent basis terms to the admittance sub-circuits. Indeed, by looking at the worst cases (bottom of Table 7.1), actually these effects are increased and the time required is almost doubled with respect to the parameter-free synthesis. These examples actually represent some very critical cases, with a high number of poles ($\bar{n} = 30$) and parameter-dependent basis order ($\bar{\ell}_D = \bar{\ell}_N = 4$): unfortunately, the circuit generated suffers in terms of speed. Nevertheless, the accuracy of the resulting netlist for all the test cases is more than satisfying: the numerical results are actually better than the one obtained from all the others synthesis.

		NoPar	GlobalPar	IndPar	CtrlPin
Case 1	Time (s)	0,518	0,642	0,665	0,731
	e_{rel}	1,03E-13	1,31E-13	1,31E-13	3,61E-14
	Matrix	66	66	66	74
Case 2	Time (s)	0,132	0,236	0,23	0,216
	e_{rel}	2,70E-14	2,07E-14	2,07E-14	1,78E-14
	Matrix	42	42	42	48
Case 3	Time (s)	0,21	0,266	0,343	0,407
	e_{rel}	4,45E-13	4,38E-13	4,38E-13	2,03E-13
	Matrix	90	90	90	102
Case 4	Time (s)	0,401	0,563	0,61	0,686
	e_{rel}	1,77E-13	1,82E-13	1,82E-13	1,56E-13
	Matrix	78	78	78	88
Case 5	Time (s)	0,238	0,41	0,342	0,422
	e_{rel}	8,74E-14	8,28E-14	8,28E-14	8,34E-14
	Matrix	58	58	58	68
Case 6	Time (s)	0,155	0,347	0,341	0,408
	e_{rel}	4,65E-14	2,93E-14	2,93E-14	2,47E-14
	Matrix	50	50	50	60
Case 7	Time (s)	0,188	0,343	0,36	0,391
	e_{rel}	3,70E-14	2,41E-14	2,41E-14	1,96E-14
	Matrix	50	50	50	60
Case 8	Time (s)	0,53	0,702	0,657	0,796
	e_{rel}	2,50E-13	3,69E-13	3,69E-13	3,35E-13
	Matrix	98	98	98	104
Case 9	Time (s)	0,561	0,721	0,689	0,768
	e_{rel}	1,11E-13	1,36E-13	1,36E-13	1,32E-13
	Matrix	98	98	98	104
Case 10	Time (s)	0,593	0,689	0,687	0,797
	e_{rel}	3,06E-13	3,08E-13	2,60E-13	1,14E-13
	Matrix	98	98	98	104
Case 11	Time (s)	0,513	0,671	0,657	0,766
	e_{rel}	3,08E-13	3,41E-13	3,41E-13	3,36E-13
	Matrix	98	98	98	104
Case 12	Time (s)	1,128	1,561	1,498	2,125
	e_{rel}	7,43E-13	7,57E-13	7,57E-13	6,92E-13
	Matrix	170	170	170	186
Case 13	Time (s)	1,047	1,313	1,345	1,702
	e_{rel}	1,28E-13	1,68E-13	1,68E-13	1,68E-13
	Matrix	146	146	146	162

		NoPar	GlobalPar	IndPar	CtrlPin
Case 14	Time (s)	1,014	1,317	1,248	1,61
	e_{rel}	6,02E-12	8,14E-12	8,14E-12	8,00E-12
	Matrix	142	142	142	154
Case 15	Time (s)	1,942	2,563	2,483	3,658
	e_{rel}	1,44E-12	1,64E-12	1,52E-12	1,09E-12
	Matrix	162	162	162	186
Case 16	Time (s)	0,468	0,782	0,796	0,984
	e_{rel}	2,67E-13	2,76E-13	2,76E-13	2,72E-13
	Matrix	202	202	202	212
Case 17	Time (s)	0,528	0,767	0,797	0,938
	e_{rel}	5,84E-13	3,67E-13	3,67E-13	2,16E-13
	Matrix	202	202	202	210
Case 18	Time (s)	1,971	3,298	3,221	4,56
	e_{rel}	3,52E-12	1,21E-11	1,21E-11	2,44E-12
	Matrix	290	290	290	302
Case 19	Time (s)	5,373	9,131	8,97	13,249
	e_{rel}	1,02E-12	2,24E-12	2,24E-12	5,63E-14
	Matrix	434	434	434	446
Case 20	Time (s)	11,923	18,505	18,636	28,334
	e_{rel}	1,07E-12	3,64E-12	3,64E-12	4,95E-14
	Matrix	578	578	578	590
Case 21	Time (s)	21,467	34,574	34,802	54,185
	e_{rel}	1,30E-12	1,64E-11	1,64E-11	4,57E-14
	Matrix	722	722	722	734

Table 7.1: Comparison of validation circuit simulations for the Test Cases of Appendix A: synthesis results with the parameter call strategies of Chapter 5. For each example we report: the overall elapsed time (Time) required to simulate all the ports for all parameter values available from the raw data, its value is obtained by adding all the 'elapsed time' fields from the log file generated by SPICE; the maximum relative error (e_{rel}) on all the available points, for all the parameter values and port responses, which is estimated by comparing the responses of the Matlab model and the SPICE simulations results; maximum matrix dimension of the circuit (Matrix), acquired by reading the 'matrix' field from the log file generated by SPICE.

7.2.2 Parametric Components Synthesis Results

We now compare the results of the synthesis of Chapter 6 based on several parameter-dependent components. Table 7.2 shows the results of the AC simulations to validate all the resistors-based syntheses (Section 6.2.1) and the VCCS-based syntheses (Section 6.2.2), by using an Independent Parameter Call strategy.
We recall that the results of Table 7.1 were obtained using a parameter-dependent VCCS component, which receives each parameter-dependent basis term as input and recomputes internally the Chebychev polynomial: the justifications of this choice are in the following.

We start analysing the implementations based on parameter-dependent resistors.
The two approaches lead almost to the same results in terms of simulations accuracy: this assumption holds also for the simulation time required. Nevertheless, they both present a drastic growth of the matrix dimension, which is directly related to the increasing number of components both for the parameter-dependent basis order and for the number of model poles. By comparing the first two columns of Table 7.2 with the last one, indeed, the major improvement provided by the parameterized VCCS elements is evident.

Some observations on the VCCS strategies can be formulated by reading data from the table. In terms of model resulting speed, the implementations of strategies (a) and (b) with a VCCS compromise the parameter-dependent netlists efficiency, also for a very small test case. Furthermore, we were not able to validate the last examples of Table 7.1 due to the massive time required by these two synthesis approaches. Moreover, the strategies (a) and (b) suffer both from a significant increment of the matrix dimension and from a considerable degradation of accuracy too.

For these reasons, we abandoned the first four strategies of Table 7.2, and we further developed only the last one, by extending it to a multivariate case and to the Control Pin call.

		Resistor (a)	Resistor (c)	VCCS (a)	VCCS (b)	VCCS (c)
Case 1	Time (s)	0,83	0,785	2,502	2,236	0,665
	e_{rel}	2,73E-13	2,66E-13	2,24E-12	2,24E-12	1,31E-13
	Matrix	128	128	198	198	66
Case 2	Time (s)	0,207	0,219	0,252	0,252	0,23
	e_{rel}	3,10E-13	3,10E-13	5,03E-13	5,03E-13	2,07E-14
	Matrix	68	68	102	102	42
Case 3	Time (s)	0,421	0,425	1,844	1,58	0,343
	e_{rel}	1,20E-12	1,20E-12	4,75E-12	4,75E-12	4,38E-13
	Matrix	188	188	294	294	90
Case 4	Time (s)	0,781	0,74	7,504	7,314	0,61
	e_{rel}	6,52E-13	6,52E-13	9,75E-13	9,87E-13	1,82E-13
	Matrix	158	158	246	246	78
Case 5	Time (s)	0,449	0,372	1,56	1,436	0,342
	e_{rel}	6,39E-13	6,46E-13	6,16E-13	6,11E-13	8,28E-14
	Matrix	108	108	166	166	58
	Time (s)	0,363	0,354	0,999	0,861	0,341

		Resistor (a)	Resistor (c)	VCCS (a)	VCCS (b)	VCCS (c)
Case 6	e_{rel}	1,66E-13	1,66E-13	2,43E-12	2,43E-12	2,93E-14
	Matrix	88	88	134	134	50
Case 7	Time (s)	0,344	0,379	0,969	0,888	0,36
	e_{rel}	1,25E-13	1,24E-13	2,05E-12	1,90E-12	2,41E-14
	Matrix	88	88	134	134	50
Case 8	Time (s)	0,986	0,906	14,219	14,065	0,657
	e_{rel}	6,63E-13	6,63E-13	4,87E-12	4,87E-12	3,69E-13
	Matrix	208	208	326	326	98
Case 9	Time (s)	0,924	0,977	13,895	13,434	0,689
	e_{rel}	5,13E-13	5,13E-13	4,49E-12	4,49E-12	1,36E-13
	Matrix	208	208	326	326	98
Case 10	Time (s)	0,875	0,874	14,266	14,346	0,687
	e_{rel}	4,20E-13	4,20E-13	7,70E-12	7,70E-12	2,60E-13
	Matrix	208	208	326	326	98
Case 11	Time (s)	0,925	0,953	13,684	13,687	0,657
	e_{rel}	9,07E-13	9,07E-13	1,80E-11	1,80E-11	3,41E-13
	Matrix	208	208	326	326	98
Case 12	Time (s)	1,232	1,139	90,01	89,407	0,796
	e_{rel}	7,87E-13	7,87E-13	4,21E-11	4,21E-11	2,76E-13
	Matrix	468	468	742	742	202
Case 13	Time (s)	1,176	1,141	83,267	82,249	0,797
	e_{rel}	7,67E-13	7,67E-13	5,47E-11	5,47E-11	3,67E-13
	Matrix	468	468	742	742	202
Case 14	Time (s)	6,97	7,204	1482,333	1448,505	3,221
	e_{rel}	1,52E-11	1,52E-11	3,27E-11	3,27E-11	1,21E-11
	Matrix	902	902	1530	1530	290

Table 7.2: Comparison of validation circuit simulations for the Test Cases of Appendix A for all synthesis types with parametric components detailed in Chapter 6. For each example we report: the overall elapsed time (Time) for the validation circuit simulations; the maximum relative error (e_{rel}) for all the port responses and parameter values; the maximum circuit matrix dimension (Matrix). See Table 7.1 for details of these fields.

7.3 Bias-Dependent Components

We now apply the proposed procedures to commonly used Circuit Blocks (CB) that are specifically designed to operate as linearly as possible around a given operating point. Such components, that are usually non-linear elements, can be tuned by changing bias conditions: therefore, a small-signal response can be obtained through full transistor-level circuit simulations or direct measurements, for model extraction purposes.
The dynamic behaviour of these components can be approximated as linear state-space systems, whose state space matrix depend on an external parameter, which in this case is the nominal bias voltage. Thus, the proposed macromodeling approach can be applied and model stability can be guaranteed. Furthermore, we are now able to realize an equivalent SPICE circuit that can receive the bias condition as a direct input.

The test cases that we are going to document are provided by an industrial partner (Intel). Some of these structures were originally proposed in [29].
Next sections present the numerical results concerning a single NMOS transistor, a two-stage buffer and more complex active analog blocks, in particular an Operational Amplifier (OA) and a Low Drop-Out Voltage Regulator (LDO). Due to their simplicity, the first two examples are inserted as a proof of the effectiveness of the proposed procedure, while the last two tests represent some more interesting cases: their possible applications in SI/PI simulations may have a direct impact on SoC architectures design.

7.3.1 NMOS Transistor

We now investigate a single NMOS transistor, originally presented in [29]. This test case structure is depicted in Figure 7.1: port one is the drain, port two the gate, and port three the bulk. The NMOS free parameter is the source-drain bias voltage $V_{ds} \in [0.8,1.2]\,\mathrm{V}$: whereby a linear region of the component characteristic is explored and a linearized macromodel of it can be generated.

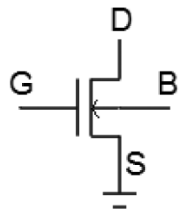


Figure 7.1: Single NMOS transistor schematic

The available dataset is composed of 41 sets of scattering responses, with $\bar{k} = 239$ frequency samples spanning the band $[1\,\mathrm{Hz}, 0.1\,\mathrm{THz}]$. The unrealistic frequency range, which is obtained through transistor-level simulations, enables to enforce the DC-point with a sufficient accuracy, necessary for a proper system linearization.
The model has been fitted with half of the datasets, while the others were used for validations, by imposing a dynamic order $\bar{n} = 4$. About the parameter-dependent basis, we selected a Chebychev polynomial order of $\bar{\ell}_N = 5$ and $\bar{\ell}_D = 4$ for numerator and denominator, respectively.

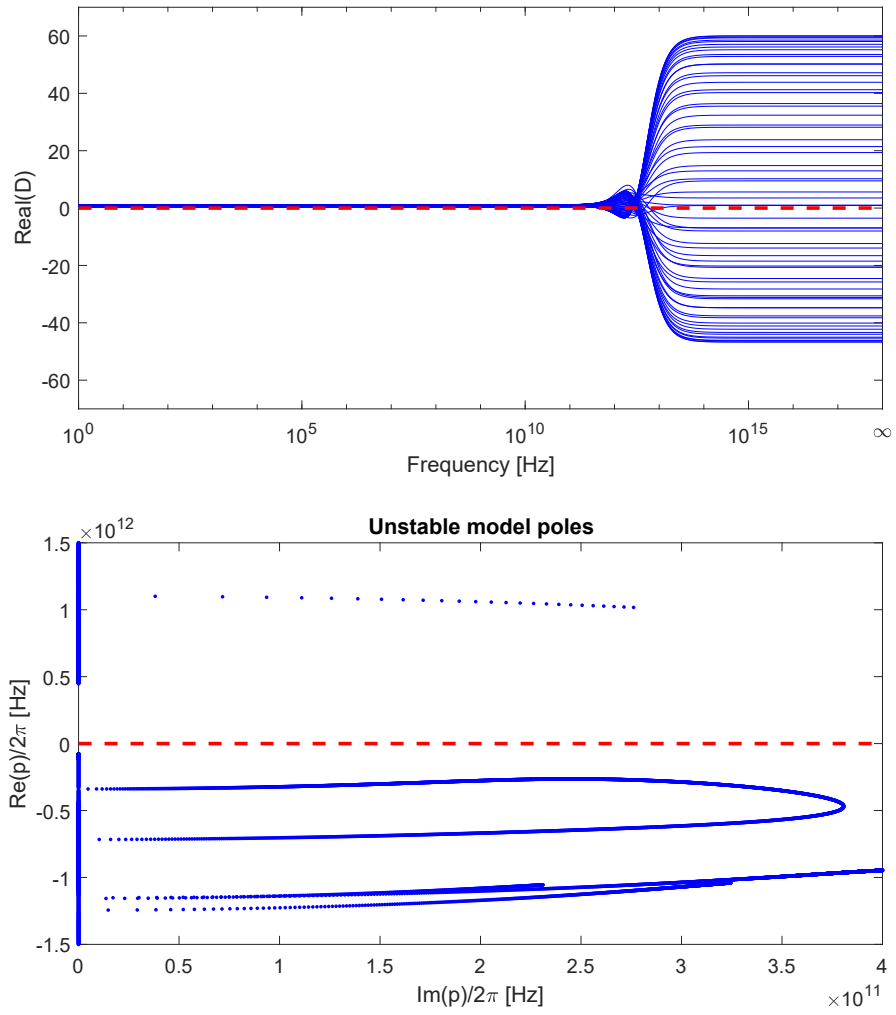


Figure 7.2: Single NMOS transistor: unstable model extracted with a standard FPSK. Top panel shows the real part of the model denominator, computed over a fine sweep of the parameter value. Bottom panel presents the unstable model poles.

Figure 7.2 presents the results from a standard FPSK procedure, for which a no-PR denominator (top panel) corresponds actually to an unstable model (bottom panel) poles. By starting from the same dataset, a new extraction has been realized in order to guarantee, through the proposed fitting procedure, the uniform model stability. Figure 7.3 shows the effectiveness of the the robust method for the model generation presented in Chapter 3. Indeed, a standard passivity enforcement on the model denominator it is not able to converge to a stable solution in a finite number of iterations: the top panels of Fig. 7.3 show that, after 20 enforcement iterations, the resulting denominator it is not yet positive-real (left panel) and that some unstable (complex) model poles are still present (right panel). By imposing a maximum number of predictive iterations $\gamma = 3$, a

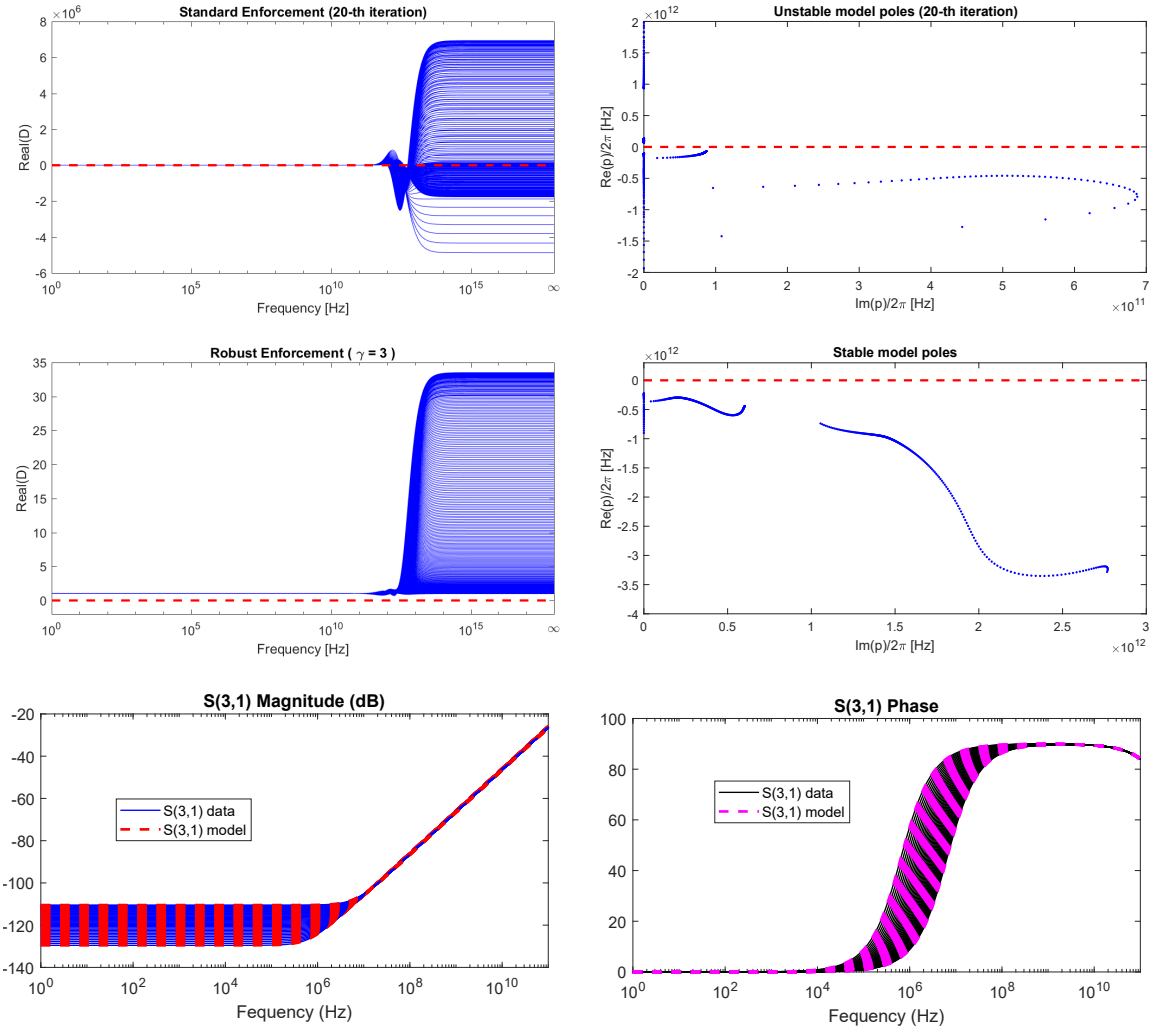


Figure 7.3: Single NMOS transistor: guaranteed stable model extraction.
Top Panels show the results from a FPSK fitting procedure with (standard) PR enforcement: (left) real part of the model denominator, computed over a fine sweep of the parameter value; (right) unstable model poles after the 20-th iteration of stability enforcement. Middle Panels report the results from a FPSK identification with a robust implementation of the final stability enforcement. In the last case, a stable model is obtained after 3 predictive iterations: since $\mathrm{Re}\{\mathrm{D}(\mathrm{j}\omega, \vartheta)\} > 0$ (see left panel), it is guaranteed that the model poles (right panel) are stable over $\vartheta \in \Theta$. Bottom panels, small-signal parameterized model responses for the port S(3,1) comparing to the corresponding real system responses, for different value of the free parameter $V_{ds} \in [0.8,1.2]\,V$. These responses are obtained from a guaranteed stable resulting model.

stable resulting model is generated with only one passivity enforcement iteration μ (for details see Section 3.4), as it is demonstrated in the middle panels of Fig. 7.3. The parameterized small-signal model responses are reported in the remaining bottom panels of

the same Figure. The last model extraction has required only 10.2 seconds, against the 34.5 seconds spent for the generation of a non-PR model with a standard enforcement. To conclude, this last example confirms the efficiency of a predictive approach, whose benefits in terms of algorithm speed-up and final accuracy are evident.

7.3.2 Two-Stage Buffer

We now consider a two-stage buffer, originally presented in [29] and depicted in the scheme of Figure 7.4. The three ports structure depends on two free parameters: the supply voltage $V_{dd} \in [0.5,1.5]\,V$, which is used to the component linearization, and the ambient temperature $T \in [20,40]\,°\mathrm{C}$.

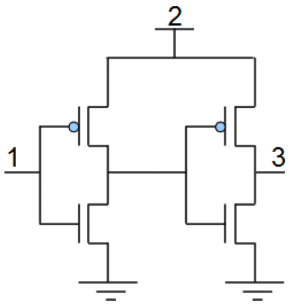


Figure 7.4: Two-Stage Buffer internal structure

The number of points along the two parameters composed the available scattering datasets: 11 sets of frequency responses are obtained along V_{dd} and 21 along T. Each one of them includes $\bar{k} = 293$ frequency samples in the band [0, 100] THz. The unrealistic extended frequency range is strictly related to the use of the resulting macromodel for transistor-level simulations, while the DC point is necessary to impose the correct bias-conditions. The model has been fitted with partial fraction basis of order $\bar{n} = 6$ and Chebychev polynomials with equal orders for denominator and numerator in both the parameter-dependent basis cases, read as $\bar{\ell}_{1,D} = \bar{\ell}_{1,N} = 2$ and $\bar{\ell}_{2,D} = \bar{\ell}_{2,N} = 3$.
The interesting result obtained in this case is that a guaranteed stable model is extracted also with a standard fitting procedure. Indeed, a standard Fast-PSK-iteration generates a model for which the denominator is positive real and, since $\mathrm{Re}\,\{\mathrm{D}(\mathrm{j}\omega, V_{dd}, T)\} > 0$, it is guaranteed that the model poles are stable over the two free parameters domains.
Even if the constrained fitting procedure proposed is not applied, this result can be achieved only with a proper check of the immittance denominator passivity, which was developed for the enforcement procedure (for details see Section. 3.1.1): its general effectiveness enables to verify (and guarantee) the stability of any case, independently by the procedure adopted for the model extraction. The results are reported in Figure 7.5.

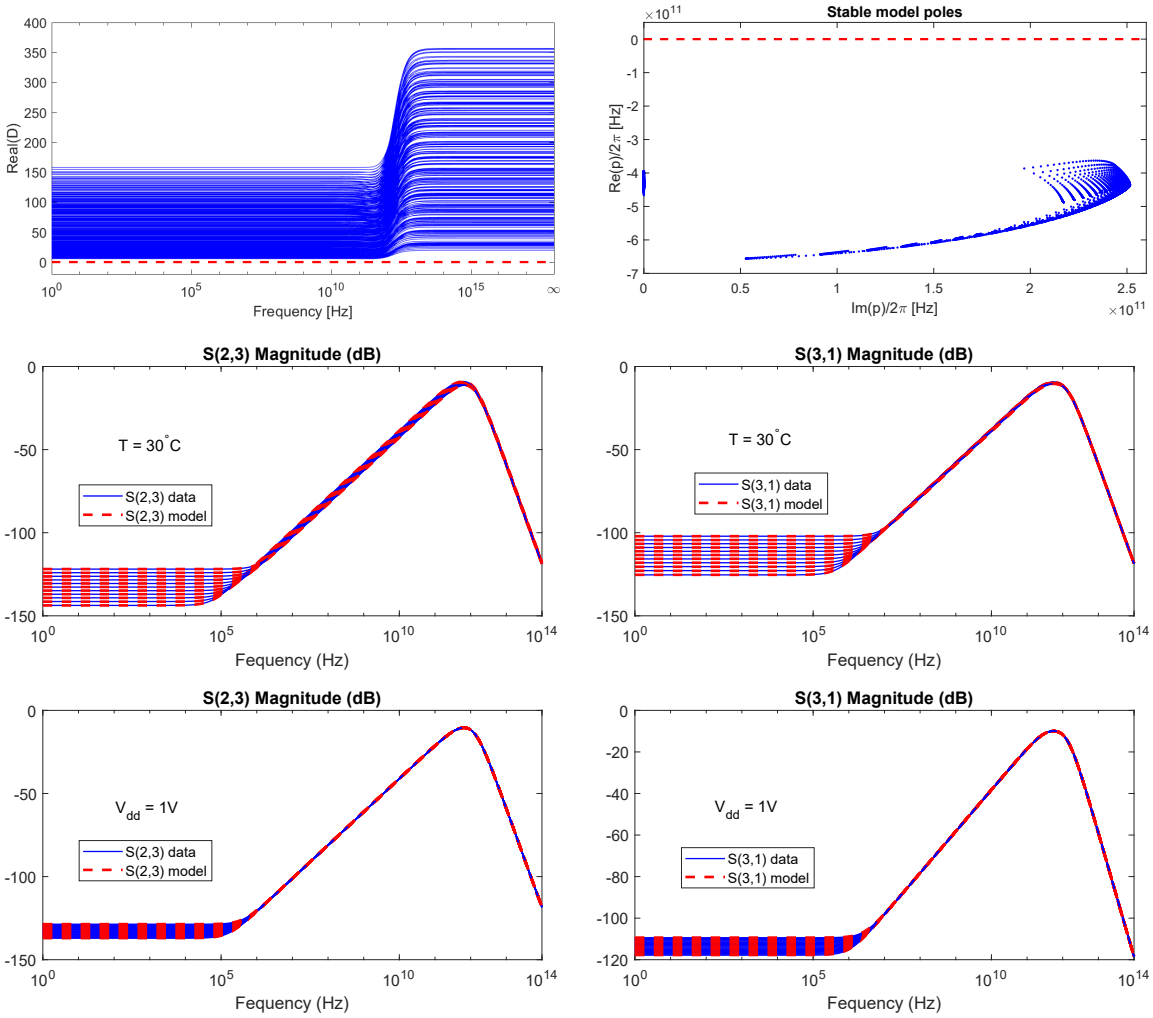


Figure 7.5: Parameterized (bivariate) model of a two-stage buffer, extracted with a standard FPSK procedure. The novel contribution provided by the (immittance) denominator passivity check enables to guarantee the model stability.
Top Panels: (left) real part of the denominator model response, which is computed through a fine sweep of both parameters values. Since $\mathrm{Re}\{\mathrm{D}(\mathrm{j}\omega, V_{dd}, T)\} > 0$, it is guaranteed that the model poles (right) are stable over the two free-parameters domains. Middle and Bottom Panels: small-signal parameterized model responses for the ports S(2,3) and S(3,1) compared to the corresponding real system responses, by fixing one parameter ($T = 30\,^{\circ}\mathrm{C}$ in the middle and $V_{dd} = 1\,V$ in the bottom) and by validating the model along the other one.

7.3.3 Operational Amplifier

The structure of this case is an Operational Amplifier (OA), part of a based-band receiver chain originally presented in [29]. The two free-parameters that parameterized the OA are its bias voltage $\vartheta_1 \in [1.1,\ 1.3]$ V and its (programmable) gain $\vartheta_2 \in [1.01,2]$.
The available dataset is composed of the scattering responses for 5 points along the first parameter and 11 along the second one. For a fixed combination of the parameters, we have $\bar{k} = 831$ frequency samples that span the band [2 Hz,20 THz]. This widespread (and unfeasible) frequency range is necessary for full transistors level simulations, and it is provided by a direct transistor-level simulation.
A stable and accurate model of the OA has been fitted, by imposing a partial fraction basis order of $\bar{n} = 14$. About the parameter-dependent bases, Chebychev polynomials have been used for numerator and denominator, of orders $\bar{\ell}_{1,N} = 2$ and $\bar{\ell}_{1,D} = 1$ for the first parameter, and $\bar{\ell}_{2,N} = 3$ and $\bar{\ell}_{2,D} = 2$ for the second one.
Therefore, in this case, to enforce the denominator passivity during the FPSK-iteration is a necessary condition to obtain a guarantee of the model stability along the parameters domains. Figure 7.6 illustrates the above mentioned results.

7.3.4 Low Drop-Out Voltage Regulator

The last bias-dependent structure that we investigate in this work is a Low Drop-Out (LDO) Voltage Regulator, originally presented in [29]. The main purpose of this component, which is taken from a mobile platform design, is to reduce the power supply noise. Indeed, through a filter mechanism, that includes the OA described in the previous section, and a feedback circuit the (output) power supply signal is 'purified' (as much as possible) of the noise inserted by the global power distribution network.
The structure is depicted in the high-level scheme of Figure 7.7: port 1 is the 'noisy' power supply port, pot 2 is the reference voltage, and port 3 is where the 'noise free' supply voltage is provided.
The LDO is parameterized through its supply voltage $V_d \in [1.2,\ 1.6]$ V. The available datasets are composed of 41 sets of scattering responses, including $\bar{k} = 802$ frequency samples spanning the band $[0\,\text{Hz}\,,\ 0.1\,\text{THz}\,]$. An extended (and unfeasible) frequency range is directly related to the purpose of comparing the macromodel behaviour with a full-transistors level simulations.
To generate a stable an accurate model, 21 of the available sets have been used for the fitting while the others have been left for validation purposes. By setting a dynamic order of $\bar{n} = 22$ and Chebychev polynomial orders of $\bar{\ell}_N = 3$ and $\bar{\ell}_D = 6$ for the numerator and denominator, respectively, the results of Figure 7.8 have been achieved. In particular, enforcing the denominator passivity during the fitting process is a necessary condition to obtain a final guaranteed stable model.

The resulting linearized behavioural macromodel is then passed through the automated SPICE extraction process, proposed during this work. The resulting parameterized netlist will be used, with the appropriate DC corrections, for transistor level simulations, by substituting the non-linear circuit component in SI/PI simulations.

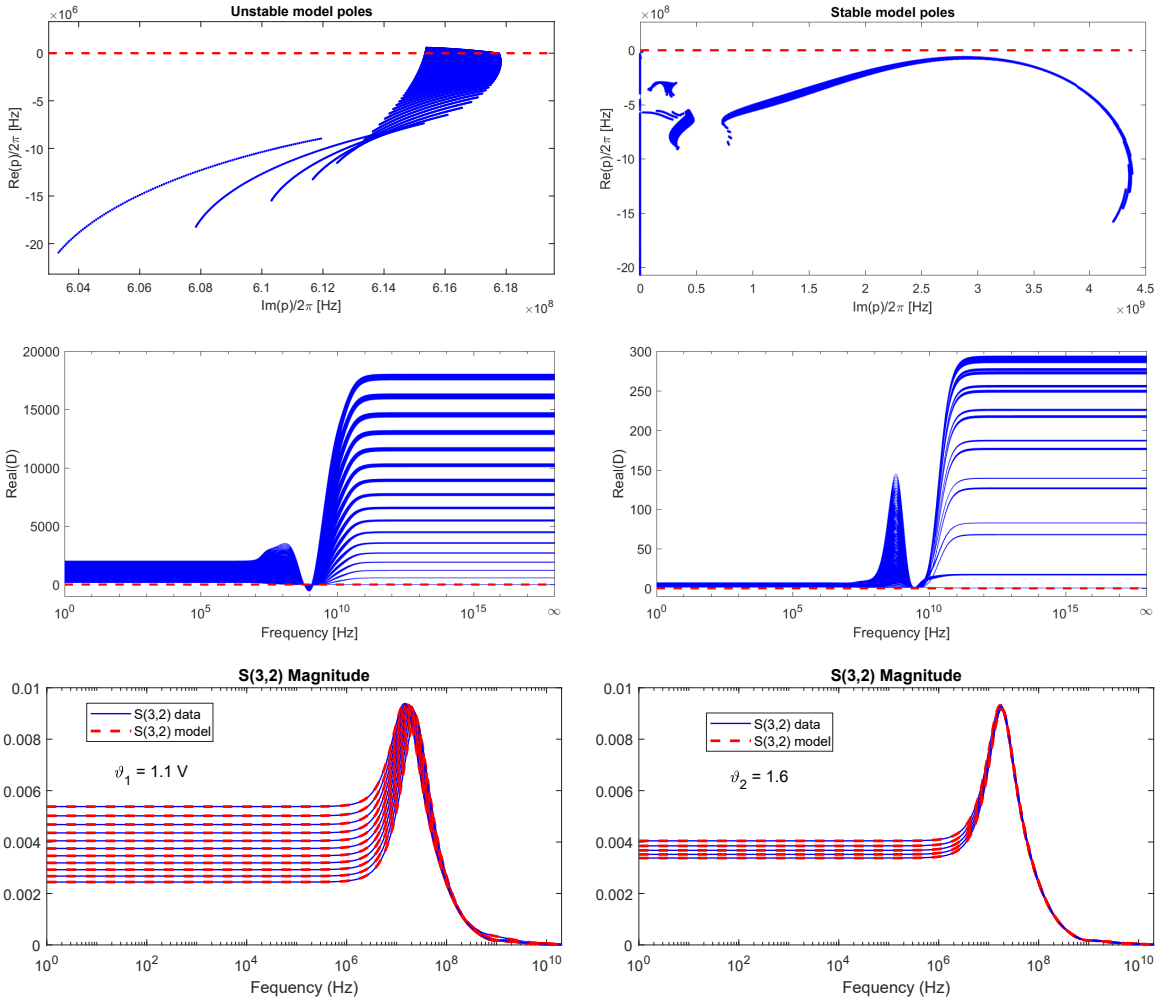


Figure 7.6: Operational Amplifier.
Top Panels: (left) unstable model poles from a standard FPSK extraction; (right) stable model poles, obtained using the proposed generation procedure. Middle Panels, real part of the model denominator evaluated through a fine sweep along the parameters: (left) model obtained through a standard FSPK-iteration; (right) model extracted by imposing the denominator passivity. Since the curve never crosses the zero baseline, the denominator is defined positive real and all the model poles are guaranteed stable.
Bottom panels: small-signal parameterized model responses compared to the corresponding real system responses, fixing one parameter at a time, for different values of the bias voltage $\vartheta_1 \in [1.1, 1.3]$ (right panel) and by making vary the gain $\vartheta_2 \in [1.01,2]$ (left panel).

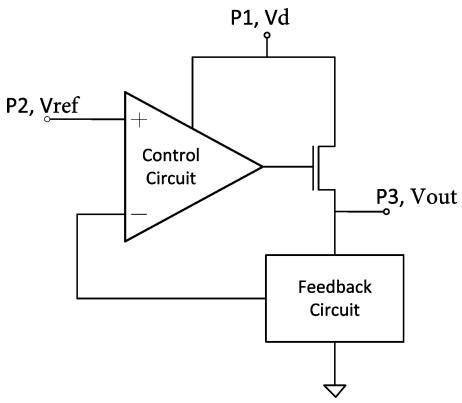


Figure 7.7: High level scheme of a Low Drop-Out Voltage Regulator.

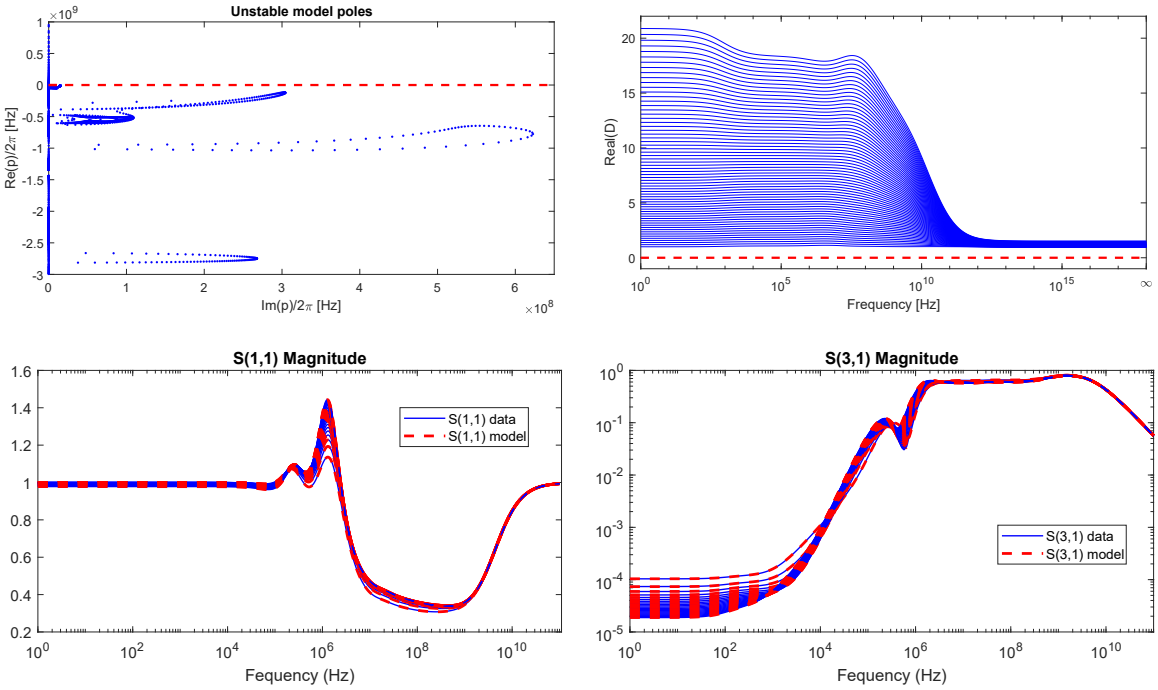


Figure 7.8: Low Drop-Out Voltage Regulator.
Top Panels: (left) unstable model poles from a standard FPSK extraction; (right) real part of the model denominator evaluated through a fine sweep along the parameters. In this case, the model is extracted by imposing the denominator passivity. Since the right result never crosses the zero baseline, the denominator is defined positive real and all the model poles are guaranteed stable.
Bottom panels: small-signal parameterized model responses compared to the corresponding real system responses, for different value of the free parameter $V_d \in [1.2, 1.6]$ V.

7.4 Function Calls

In this section, we document all the functions necessary to create the validation circuits and the SPICE model extractions, which provides the numerical results of this chapter. Depending on the synthesis procedure chosen, we will see the following:

1. LTSpiceModel2DUM_noPAR , realizes a parameter-free SPICE extraction of a standard GSK-model (see Chapter 4) and validates it on all the available points from the raw data-set, for all the port responses, with proper AC circuits.

2. LTSpiceModel2DUM_Par , realizes the equivalent SPICE netlist of a PSK-model, by using during the synthesis a parameter-dependent VCCS component. Proper AC circuits enable to validate the netlist over all the parameters values and port responses. The parameter call strategy can be selected with the specific option field (see Chapter 5).

3. LTSpiceModel2DUM_par_CompStrategy , is a set of functions that realizes the PSK-model extractions of Chapter 6 and validates them with proper AC circuits.

Where

- Comp is a string depending on the parameter-dependent component type required during the SPICE realization, and in particular, could be set to
  - R, if a parameter-dependent resistor is created during the model synthesis (see Sec. 6.2.1);
  - VCCS, if parameter-dependent VCCS is used for the SPICE netlist generation (see Sec. 6.2.2);

- Strategy is a string depending on the procedure used to synthesize the parameter-dependent polynomial. Recalling to Sec. 6.1, we can summarize the available ones as:
  - a , the normalized parameter is provided as an input of a dedicated sub-circuit, which evaluates each parameter-dependent basis term and recomputes the model coefficient $r_n(\vartheta)$;
  - b , a proper sub-circuit is defined in order to get as input only one parameter-dependent basis term;

In the following sections, we will document each of them, starting from the parameter-free realization. The other two points, related to the parameterized SPICE generations of Chapter 5 and Chapter 6, will be documented together due to their similarity.

7.4.1 GSK-model Synthesis Validation

This function validates a model in GSK-form with respect to the true system response $\breve{\mathbf{H}}(\mathrm{j}\omega, \vartheta)$ using LTSpice as the general framework.
This is a driver for all the functions necessary to the SPICE model extraction, which are detailed in Chapter 4: the input variables must be compliant with this statement.
The input model must be a ratio between numerator and denominator series objects,

which must be defined through the single-factor 'partialfraction' basis. The input dataset is stored in a tabulated frequency data structure that must be non-parameterized. *Options* field enables a customization of the SPICE synthesis, as detailed below.

The output files are saved in a folder named *testGSKModel_spice*, which is created in the current working folder.

A note of attention is placed here. This function is compatible only with both the MAC OS X and Windows version of LTSpiceIV: the computer operating system is automatically selected. In the last case, the user must properly insert in the variable *LTSPICE*, inside this function, a string with the path of the LTSpice execution file 'scad3.exe'. We provide an example in the following.

```
% LTSpice command location and runtime parameters
if ismac
LTSPICE = '/Applications/LTspice.app/Contents/MacOS/LTspice -b ';
else
LTSPICE = 'C:\Users\marco\Desktop\LTC\LTspiceIV\scad3.exe -b ';
end
LTSPICEOPT = ' -run ';
% Set this to 1 to run LTSPICE manually. Program will pause.
OFFLINE = 0;
```

The above procedure enables to automatically run a SPICE simulation in 'batch mode'.

Here follows the function call.

```
function [DUMSpice,DUM,TotTime,MaxMatrixSize] = LTSpiceModel2DUM_noPAR
    (Model,Data,Options)
```

Where the inputs are:

- Model is the model in the GSK-form;
- Data is the structure that stores all the frequency samples;
- Options is an (optional) input parameter that includes the fields of the function of Section 4.4.1 GSK_Model2Cir.

The function returns as output:

- DUMSpice is a structure which stores the Spice model port responses.
- DUM is a structure which stores the same frequency samples of Data in a DUM format.
- TotTime is the overall elapsed time to validate the model for all the port responses: its value is obtained as the sum of the same field from the log files generated by SPICE.
- MaxMatrixSize is the maximum size of the circuit matrix, obtained from the SPICE log files.

7.4.2 PSK-model Synthesis Validation

These functions validate a model in PSK-form with respect to the true system response $\breve{\mathbf{H}}(\mathrm{j}\omega, \vartheta)$ using LTSpice as working environment.
This is a driver for all the functions necessary to the parameterized SPICE model extractions, which are detailed in Chapter 5 and Chapter 6: the input variables must be

compliant with this statement.
In particular, the input model must be a ratio between numerator and denominator series objects, which must be defined through the 'partialfractions' and 'chebychev' bases. The input parametric dataset is stored in a tabulated frequency data structure. The *Options* field enables a customization of the SPICE synthesis, as detailed in Section 5.6.1.

The output files are saved, for the second function, in a folder named *cktname_*PARtype, which is created in the current working folder. PARtype is a string depending on the parameter call type required. The set of functions output folder name is selected according to the synthesis strategy used as *testGSKModel_spice_PAR_*CompStrategy: we refer to the introduction of this section for details.

A note of attention is placed here. This function is compatible only with both the MAC OS X and Windows version of LTSpiceIV: we refer to Section 7.4.1 for details.
Here follows the functions call

```
function [DUMSpiceList,DUMList,TotTime,MaxMatrixSize] =
    LTSpiceModel2DUM_Par(Model,Data,Options,cktname)

function [DUMSpiceList,DUMList,TotTime,MaxMatrixSize] =
    LTSpiceModel2DUM_par_Ra(Model,Data,Options)
function [DUMSpiceList,DUMList,TotTime,MaxMatrixSize] =
    LTSpiceModel2DUM_par_Rc(Model,Data,Options)
function [DUMSpiceList,DUMList,TotTime,MaxMatrixSize] =
    LTSpiceModel2DUM_par_VCCSa(Model,Data,Options)
function [DUMSpiceList,DUMList,TotTime,MaxMatrixSize] =
    LTSpiceModel2DUM_par_VCCSb(Model,Data,Options)
```

The inputs are:

- Model is the model in the PSK-form;
- Data is the structure that stores all the frequency and parameters samples;
- Options is an (optional) input parameter that includes the fields of the function of Section 5.6.1 GSK_Model2Cir_Par.
- cktname is an (optional) input which stores the SPICE sub-circuit name in the output file *'name.cir'* (Do not include the extension in the input string). This is also the first part of the output folder name, which is created as *cktname_*PARtype. If this input is not provided, the variable is initialized as *'PAR_GSKmodel'*.

The function returns as output:

- DUMSpice is an array of structures that store the Spice model port responses for all the parameter values.
- DUM is an array of structures structure which stores the same frequency samples of Data in a DUM format for each parameter value.
- TotTime is the overall elapsed time to validate the model for all the parameter value and port responses: its value is obtained as the sum of the same field from the log files generated by SPICE.
- MaxMatrixSize is the maximum size of the circuit matrix, obtained from the SPICE log files.

Chapter 8

Conclusions and Further Improvements

This dissertation work proposed a behavioural macromodeling approach to create surrogate parameterized models that are guaranteed stable through the constrained identification process, which is presented in Chapter 3. By enforcing the (immittance) denominator passivity at each iteration of the fitting procedure, we are able to realize a positive-real denominator, which is a sufficient condition to guarantee that all model poles are stable in the parameter domain. A second important result has enabled to automatically synthesize a parameterized SPICE surrogate of the model previously extracted (we refer to Chapter 5 and Chapter 6), in order to take advantage of the order reduction procedure during system-level verification of signal and power integrity (SPI). Indeed, the generated equivalent SPICE networks can be applied to the design process, by varying the design parameters through numerical simulations. The above procedures have been applied to a particular class non-linear systems, designed to work in a (linear) characteristic region that depends on the given operating point. These bias-dependent components (such as NMOS transistors, I/O buffers, low-noise Operational Amplifiers and Low Drop-Out Voltage Regulators) have been parameterized by their nominal bias-voltage and corresponding stable macromodels have been generated and validated. These results fulfilled the initial thesis goals, even if further improvements could strengthen the proposed work.

Reduced Set of Constraints for the Stability Enforcement

About the stability enforcement, the strategy based on a positive-real denominator shows a general effectiveness for the model extraction, both in terms of required time-consuming and resulting accuracy with respect to the true response. Nevertheless, the algorithm performance can be enhanced by modifying the construction of the feasibility set, which is realized at each iteration of the fitting process. Indeed, up to now the problem constraints are formulated starting from all available points of the raw dataset and adding to them the passivity violations points (for details see Chapter 3): this approach results in large and memory-demanding matrix constraints.
A first improvement of the algorithm requires optimizing the set of constraints, in order to approximate the optimization problem feasibility region with a minimum number of

constraints. By creating a reduced equivalent set of constraints at each iteration of the identification routine, a general speed-up of the entire process is expected without any impact on the resulting model.

Fully Automated Model Extraction

Another improvement that can be realized to strengthen the proposed model generation is to reduce the user impact on the extraction, by realizing a fully automated procedure. At the present day, the resulting model accuracy is strongly affected by the choice of bases functions orders, both for the frequency and parameter: this selection must be performed by the user, which has to manually change the several starting options to realize a model that satisfies the requirements. This strategy, which is based on a visual inspection of the validation results, does not assure that the final model will be the best one among the set of possible configurations. Thus, several attempts are performed by the user in order to obtain a "good enough" resulting model: this tedious procedure (set the options, extract a model and validate it) reveals an intrinsic time-consuming nature, which cannot be reduced despite the experience that the user has accumulated performing this work.

A solution to this issue is not straightforward, and some advanced approaches are required. In particular, one possible strategy is to implement an automatic extraction based on a neural network. This can be created in order to 'guess' an optimum starting configuration of the model by analysing the raw dataset, both along the frequency and the parameters directions. How to train the neural network or how to realize a widespread (and sufficient) set of test cases are still open issues that are left for further investigations.

SPICE Synthesis Applications

About the parametric SPICE equivalent synthesis, few more 'improvements' can be realized to this work. Indeed, the automatic tool for the netlists extraction has been tested only with LTSpice. A possible extension of it to any commonly used circuit solver (PSpice, HSpice, Spectre) is suggested to improve the tool applicability to an industrial environment.

Moreover, an interesting but unexplored application for a parameter-dependent netlist is strictly related to the use of a Control Pin (for details see Sec. 5.5). Indeed, a self-parametrized SPICE model of a bias-dependent component could be realized by creating a proper filtering circuit. This could provide the constant part of the input signal as the external parameter on the proper model pin, while the circuit block behaviour can be studied when a variable noisy signal is provided as input. Nevertheless, this application requires adapting the parameter-dependent DC correction, originally presented in [29], which must be applied to the parameterized model. This procedure is not yet available but could be the subject of a future investigation.

Appendix A

Test cases

We detail in the following all the test cases used throughout the proposed work as numerical examples. In particular, for each one of them, we introduce the physical structure under test and the fitting algorithm settings necessary to generate the presented results. We focus in particular both on the basis functions orders and the datasets selected, due to their impact on model extraction.
The fitting procedure used as a standard in the below examples is the one proposed in Chapter 3 and the stability enforcement is embedded at each fitting iteration: when this statement is not true, it is specified in the test case description.

Case 1

Microstrip Filter with Double Folded Stub
A microstrip band-stop filter with double folded stubs is the first test case considered during the proposed work. This physical structure, depicted in Figure A.1, was presented in [10] and [38].

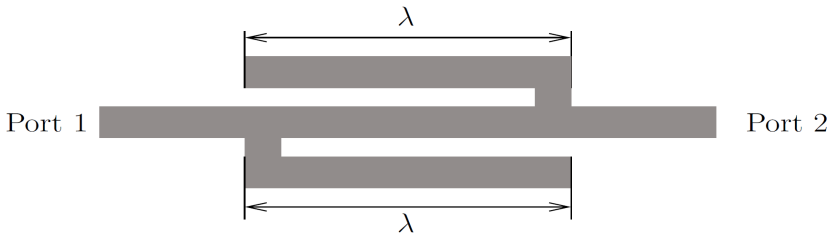


Figure A.1: Microstrip Filter with Double Folded Stub

The above structure is parameterized by the length of the folded stubs $\lambda \in [2.08,\ 2.28]$ mm. The behaviour of the filter is characterized by 21 datasets of scattering responses, including $\bar{k} = 300$ frequency samples over the band [5, 20] GHz. Only 11 of the available sets were used to fit the model: the remaining 10 are used to validate it.
A guaranteed stable model is extracted by imposing a partial fractions basis order $\bar{n} = 10$

and by using Chebychev polynomials as parameter basis, with the common order $\bar{\ell} = 2$ both for the numerator and denominator. The maximum errors among the model and validation data is $4.47 \cdot 10^{-3}$.

Case 2

This test case is a commonly used PCB integrated capacitor, which physical structure characteristics (such as the dielectric material, the number of layers or its geometry) are not provided. The free parameter selected for this case is the side length of the component that varies in the range [254,609.6] μm. The available true response dataset is composed of 8 sets, one for each parameter value, containing 191 frequency samples that span the band [0.5,10] GHz.
In this case, a guaranteed stable model has been fitted with frequency-dependent basis function order $\bar{n} = 4$ and with Chebychev polynomial as the parameter-dependent basis, both for numerator and denominator, of order $\ell_N = \ell_D = \ell = 1$. The maximum relative error between the obtained model and validation data is $1.16 \cdot 10^{-2}$.

Case 3

Link on printed circuit board
This test case is an S-shaped link on a printed circuit board realized on an FR4 epoxy board (thickness 0.76 mm, dielectric constant 4.4 , loss tangent 0.02). The structure, 300 μm wide, was originally presented in [39] and it is represented in Figure A.2: the middle-section length represents the free parameter $L \in [2, 18]$ mm, while the other two segments are fixed at 2 cm. The structure behaviour is captured through 9 sets of scattering responses, which span the band [0.1, 10] GHz including $\bar{k} = 100$ frequency samples each.

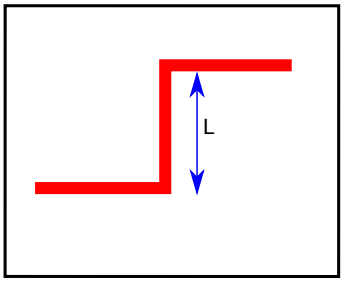


Figure A.2: Link on printed circuit board

The model to which we refer has been fitted with partial fractions basis order $\bar{n} = 16$, by imposing the denominator passivity during each fitting iteration. About parameter basis functions, we adopted Chebychev polynomial with numerator and denominator orders equal respectively to $\bar{\ell}_N = 4$ and $\bar{\ell}_D = 3$. This last choice does not leave any data to validate our final result, indeed all the available dataset are used to fit the final (stable)

model. However, the maximum relative error for all the ports and for all the available parameter values is $2.69 \cdot 10^{-1}$.

Case 4

Via with Residual Stub

This structure, depicted in Figure A.3, is a via connecting a microstrip line and a stripline in a multi-layer PCB, originally presented in [35]. This component is parameterized through the stub height h, vary in the range $[0,\ 716]\mu$m: indeed, a backdrilling procedure enables to change the stub, which is a residual of the metallization process that realizes the via and that may cause SI (signal integrity) issues.

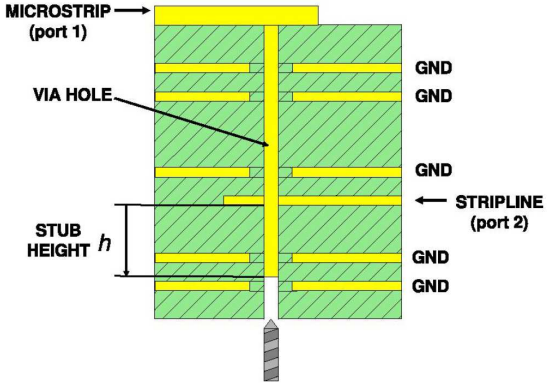


Figure A.3: Via with residual stub

The dataset is composed of 10 set of scattering responses, including $\bar{k} = 1001$ frequency samples spanning the band [0, 40] GHz. We fitted the model with partial fractions basis order $\bar{n} = 13$ and with Chebychev polynomials as parameter basis functions, with orders $\bar{\ell}_N = 3$ and $\bar{\ell}_D = 3$ for numerator and denominator, respectively. We selected 8 of the available datasets for the fitting, while the other 2 are used to validate the guaranteed stable model.
The maximum relative error between the obtained model and validation data is $2.75 \cdot 10^{-2}$.

Case 5

Multi-Layer integrated Inductor

The structure under test is a spiral integrated inductor with 1.5 turns, placed on a multilayer substrate (courtesy of Prof. M. Swaminathan, Georgia Institute of Technology, Atlanta, GA, USA). The inductor, which presents a square outline, has a variable side-length in a range of [1.02, 1.52] mm, which is assumed as the design parameter.
The test case behaviour is captured in a set of 11 frequency responses: 7 of them are used to fit the model, while the others are used as validation data-sets. The model is fitted with a partial fraction basis order $\bar{n} = 8$. About the parameter-dependent basis, we used

Chebychev polynomials of order $\bar{\ell}_N = 3$ and $\bar{\ell}_D = 2$ for numerator and denominator, respectively. To guarantee a stable resulting model, after the constrained fitting process, 1 final passivity enforcement iteration on the denominator was required. The worst case relative error among all the available samples is $8.37 \cdot 10^{-2}$.

Case 6-7

Integrated Inductor
In this case, the physical structure is a PCB integrated inductor, whose number of turns (1.5 or 2) originates two different datasets. However, the 11 sets of frequency responses depend on the same free parameter, which is the side-length of the square outline of the inductor. Its value span the range of [1.02, 1.52] mm. These two test cases are presented in the following.

Case 6

Integrated Inductor with 1.5 Turns
For this example, each set of data is composed of $\bar{k} = 477$ frequency samples span the band [0.1 ,12] GHz. Of the overall 11 sets of samples available, 7 of them are selected to realize the model while the others are used for validation.
A guaranteed stable model is fitted with partial fractions basis order $\bar{n} = 6$ and Chebychev polynomial as parameter basis functions with orders $\bar{\ell}_N = 3$, $\bar{\ell}_D = 2$. To obtain a positive real denominator, one iteration of the final passivity enforcement is required. The maximum relative error between the resulting model and validation data is $2.18 \cdot 10^{-3}$.

Case 7

Integrated Inductor with 2 Turns
In this case, the 11 sets responses span the bandwidth [0.1, 19.75] GHz through $\bar{k} = 476$ frequency samples. In order to fit a guaranteed stable and accurate result, only 4 sets of available data are used for validation.
The model has been obtained with a frequency-dependent basis order $\bar{n} = 6$. About the parameter-dependent basis, a Chebychev polynomial for numerator and denominator has been selected, with orders equal respectively to $\bar{\ell}_N = 3$ and $\bar{\ell}_D = 2$.
The fitted model, which is guaranteed stable without requiring a final passivity enforcement on the denominator, presents a maximum relative error with respect to the validation data of $3.12 \cdot 10^{-3}$.

Case 8-11

Transmission Line Network with Embedded Discontinuity
This structure is a transmission line with an embedded discontinuity, modelled through an RLC circuit. This case was originally presented in [21], and is depicted in Figure A.4. The capacitance C and inductance L_1 are the model free parameters: their combinations

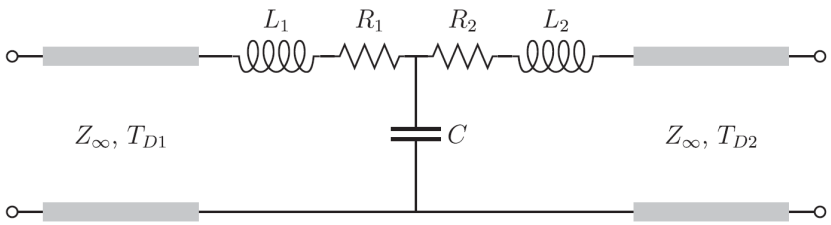


Figure A.4: Transmission Line With Embedded Discontinuity

generate the datasets detailed below. However, the others components are kept fixed to the following values

- Lines characteristic impedance $Z_\infty = 40\Omega$;
- Propagation times $T_{D1} = 100$ ps, $T_{D2} = 230$ ps;
- Discontinuity resistance $R_1 = R_2 = 1\Omega$;
- Discontinuity inductance $L_2 = 10$ nH.

The several structure behaviours are obtained through (SPICE) circuit simulations and present common characteristic: each one of them is composed of 11 sets of frequency responses, including $\bar{k} = 1000$ samples in the band $[10 \cdot 10^{-3}, 10]$ GHz. Moreover, we selected only 6 sets of each sub-dataset to fit the several stable models, while the others were used for validation purpose. Furthermore, no one of these cases has required a final passivity enforcement on the denominator to guarantee the model stability. We detail the available sub-cases in the following.

Case 8

The free parameter of this sub-case is the capacitance value $C \in [0.1,10]$ pF, while the inductance L_1 is fixed to 10 nH.
The (stable) model has been fitted with a partial fraction order of $\bar{n} = 18$ and with Chebychev polynomials (for the parameter-dependent basis) of order $\bar{\ell}_N = \bar{\ell}_D = 1$. The maximum relative error between the resulting model and validation data is $3.20 \cdot 10^{-3}$.

Case 9

For this structure, the capacitance C is selected as the free parameter but now in the range $C \in [1,10]$ pF, while the inductance L_1 is still fixed to 10 nH. The same statements made for the **Case 8**, about validation datasets and bases functions orders, still hold. The maximum relative error between the resulting model and validation data is $7.40 \cdot 10^{-4}$.

Case 10

In this case, the capacitance value varies in a range of $C \in [0.1,1]$ pF, while the inductance L_1 is fixed to 10 nH. We did the same statements made for the **Case 8** and **Case 9**, but now the maximum relative error between the resulting model and validation data is $2.33 \cdot 10^{-3}$.

Case 11

The free parameter for this test case is the inductance value $L_1 \in [0.01,1]$ nH, while the capacitance value is fixed to $C = 1$ pF. Also for this last sub-case the same statements of the above ones, about validation datasets and bases functions orders, still hold. The maximum relative error between the resulting model and validation data is $2.33 \cdot 10^{-3}$.

Case 12-15

PCB Interconnect Over a Slotted Reference Plane
This structure is a PSB microstrip on a dielectric reference surface with a rectangular discontinuity of length L and distance d from the center of the reference plane, which breaks the current return path [21]. A scheme of the physical structure is reported in Figure A.5.

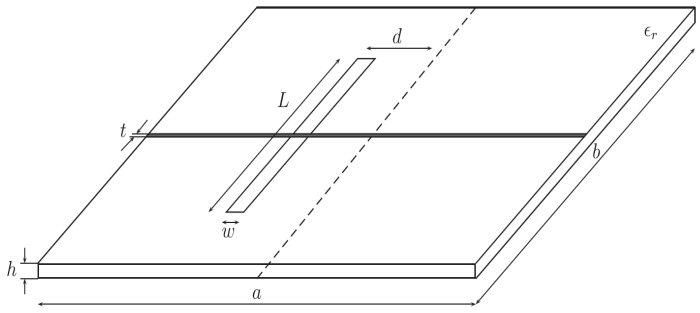


Figure A.5: PCB Interconnect Over a Slotted Reference Plane

The fixed geometrical parameters are the following

- $a = 100$ mm;
- $b = 100$ mm;
- $t = 0.035$ mm;
- $w = 0.12$ mm;
- $h = 0.3$ mm;
- $L = 25$ mm;
- $\epsilon_r = 4.7$.

The structure free parameters are the slot offset from the plane center $d \in [0, 25]$ mm and its length $L \in [1\ ,25]$ mm. Through full-wave simulations, reference scattering datasets were obtained combining the values of this two parameters. Each one of these sets is composed of $\bar{k} = 1858$ frequency samples over the band [0, 10] GHz. We are going to detail the resulting sub-cases in the following.

Case 12

In this case, the slot length is fixed L to 25 mm, while the slit offset from the plane center d is selected as the free parameter: the resulting dataset is composed of 9 sets of frequency samples. The model is fitted with partial fractions basis order $\bar{n} = 36$. Chebychev polynomials are used as parameter basis functions, with orders $\bar{\ell}_N = 10$ and $\bar{\ell}_D = 3$ for numerator and denominator, respectively. All the available sets are used to

generate the final model, which is not guaranteed stable by the fitting procedure. Indeed, this case is used to stress the equivalent SPICE extraction of a model with high bases functions order, both for the frequency and parameter: the stability is not required for such purpose. However, the maximum relative error between the resulting model and validation data is $2.45 \cdot 10^{-3}$.

Case 13

The structure has a fixed discontinuity offset $d = 0$ mm and a variable slot length L that results in a dataset of 9 elements. All the available sets are used to fit a model, which is not guaranteed stable. However, we select this case to stress the SPICE extraction procedure due to high order for both the frequency and parameter dependent bases, and we do not consider this as an issue. In particular, the bases orders are fixed to $\bar{n} = 30$ for the partial fractions basis, to $\bar{\ell}_N = 10$ and $\bar{\ell}_D = 3$ for the numerator and denominator Chebychev polynomials. The maximum relative error between the resulting model and validation data is $9.30 \cdot 10^{-3}$.

Case 14

In this case, the free parameter of the considered structure is the slot length Ls, while the lit offset d is fixed to 25 mm. All the 9 sets of frequency samples data are used to fit an accurate model, which is not guaranteed stable but that is used to stress the equivalent netlist generation procedures. Indeed, an accurate model is fitter with the partial fractions basis order $\bar{n} = 29$ and parameter-dependent basis order common for numerator and denominator of $\bar{\ell}_N = \bar{\ell}_D = 4$. The maximum relative error between the resulting model and validation data is $7.26 \cdot 10^{-3}$.

Case 15

The free parameter for this case is the discontinuity offset $d \in [0{,}25]$ mm, while the slot length is fixed at $L = 25$ mm. To fit an accurate model we used all the 9 available sets, thus no data is used to validate the final results. To generate a model, we set the partial fractions basis order to $\bar{n} = 34$. About the parameter-dependent basis, a Chebychev polynomial was used both for the numerator $\bar{\ell}_N = 10$ and for the denominator $\bar{\ell}_D = 3$. The maximum relative error between the resulting model and validation data is $1.10 \cdot 10^{-2}$.

Case 16-17

Printed Circuit Board Interconnect
This test case is a high-speed PCB signal link with two PCB's, attached by a connector, that enables the signal path through a stripline rounder in their inner layer. For details see [31] [16], [21] (Courtesy of Prof. Christian Schuster and Dr. Jan Preibisch, Technische Universität Hamburg-Harburg, Hamburg, Germany). The pad $a \in [100{,}300]$ μm and anti-pad $b \in [400{,}600]$ μm radii are the two free parameters. The true system responses are obtained through full-wave simulations and compose a dataset of 81 scattering responses with 9 elements for each parameter value. The $\bar{k} = 1000$ frequency samples span the

band [1 Hz, 20 GHz] for each set of data.
Several sub-cases are realized combining the available free parameters. We detail them in the following.

Case 16

In this case, anti-pad radii is fixed at $b = 400\mu$m while the free parameter is the pad radii $a \in [100{,}300]\,\mu$m. On the 9 sets available, only two (the fourth and sixth) are used for validation purpose, while the others are necessary for the fitting. The model is extracted with a partial fractions basis function of order $\bar{n} = 44$ and with Chebychev polynomials of orders $\bar{\ell}_N = 3$ and $\bar{\ell}_D = 2$, respectively for the numerator and denominator. The maximum relative error between the guaranteed stable model and validation data is $3.51 \cdot 10^{-2}$.

Case 17

The free parameter for this test case is the anti-pad radii $b \in [400{,}600]$ μm, indeed the pad radii is fixed to $a = 100\mu$m. The same statement about datasets of **Case 16** still holds. A guaranteed stable model has been extracted imposing an order of $\bar{n} = 44$ for the partial fractions basis functions, while the Chebychev polynomial order was kept equal for the numerator and denominator to $\bar{\ell}_N = \bar{\ell}_D = 2$. The maximum relative error between the resulting model and validation data is $3.28 \cdot 10^{-3}$.

Case 18-21

Coupled Transmission Lines
These test cases consider a set of N differential pairs of two parallel wires, placed nearby as depicted in Figure A.6 and presented in [16].

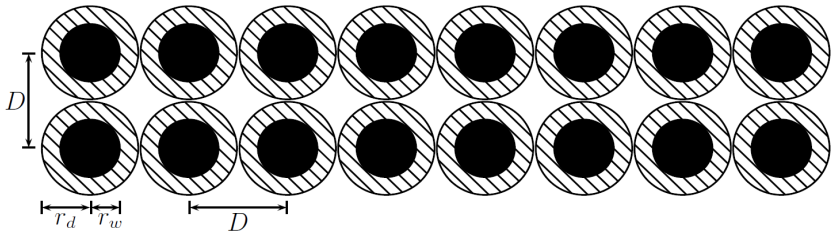


Figure A.6: Coupled Transmission Lines

The general structure characteristics are

- wires length=10cm;
- conductors radius $r_w = 0.5$ mm;
- dielectric radius $r_d = 0.8$ mm;
- relative permittivity $\epsilon_r = 4.2$;
- center wires distance $D = 1.61$ mm.

The free parameter of this structure is the coupling length $L_c \in [20, 40]$ mm, while each pair can be considered decoupled over a length $L - L_c$. This situation results in a $2N$ multi-conductor line. The scattering responses, which contain $\bar{k} = 500$ frequency samples, are obtained through full-wave solver for 11 parameter values.
By adding a pair or coupled wires N we construct the sequential sub-cases: their models are extracted by imposing the same fitting options. Indeed, we used these sub-cases to stress the SPICE extractions with an increasing number of model poles and a fixed order for the bases functions. For the same reason, the stability of this model was not enforced and all the available data were used for the fitting. Nevertheless, except **Case 18** all the others models present a PR denominator through the external parameter space, thus are guaranteed stable. In particular, we fit the models with a partial fractions basis of order $\bar{n} = 30$ and with a common Chebychev polynomial order for numerator and denominator of $\bar{\ell}_N = \bar{\ell}_D = 4$. The maximum relative errors between the resulting models and validation data are shown in TableA.1, which details also the number of wires of each sub-cases and the corresponding number of model ports.

	$max(e_{rel})$	N	P
Case 18	$1.13 \cdot 10^{-2}$	2	4
Case 19	$9.83 \cdot 10^{-3}$	3	6
Case 20	$2.12 \cdot 10^{-2}$	4	8
Case 21	$2.06 \cdot 10^{-2}$	5	10

Table A.1: Case 18 to 21: (left) maximum relative errors among resulting models and validation data for all the parameter values and ports; (center) number of coupled wires N; (right) number of model ports $P = 2N$.

Bibliography

[1] *LTSPICE IV,* Linear Technology*, available online: www.linear.com.*

[2] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions.* Dover Publications, 1968.

[3] R. Achar and M. S. Nakhla. Simulation of high-speed interconnects. *Proceedings of the IEEE*, 89(5):693–728, May 2001.

[4] B. Anderson and S. Vongpanitlerd. Network analysis and synthesis: A modern systems theory approach, eaglewood cli s, 1973.

[5] J. P. Boyd. *Chebyshev and Fourier spectral methods.* Courier Corporation, 2001.

[6] S. Boyd, V. Balakrishnan, and P. Kabamba. A bisection method for computing the h norm of a transfer matrix and related problems. *Mathematics of Control, Signals and Systems*, 2(3):207–219, 1989.

[7] T. Bradde. Fast data-driven algorithms for parameterized macromodeling of multi-port systems. Master's thesis, Politecnico di Torino, 2018.

[8] T. Chihara. An introduction to orthogonal polynomials (gordon and breach science publishers, new york, ny). Technical report, ISBN 0-677-04150-0, 1978.

[9] M. De Stefano, S. Grivet-Talocia, T. Bradde, and A. Zanco. A framework for the generation of guaranteed stable small-signal bias-dependent behavioral models. In *Microwave Conference (EuMC), 2018 European.* IEEE, 2018.

[10] F. Ferranti, L. Knockaert, and T. Dhaene. Parameterized s-parameter based macromodeling with guaranteed passivity. *IEEE Microwave and Wireless Components Letters*, 19(10):608–610, 2009.

[11] F. Ferranti, L. Knockaert, and T. Dhaene. Guaranteed passive parameterized admittance-based macromodeling. *IEEE Transactions on Advanced Packaging*, 33(3):623–629, 2010.

[12] F. Ferranti, L. Knockaert, and T. Dhaene. Passivity-preserving parametric macromodeling by means of scaled and shifted state-space systems. *IEEE Transactions on Microwave Theory and Techniques*, 59(10):2394–2403, 2011.

[13] A. Gil, J. Segura, and N. M. Temme. *Numerical methods for special functions*, volume 99. Siam, 2007.

[14] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. Johns Hopkins Univ Pr, 1996.

[15] S. Grivet-Talocia. Passivity enforcement via perturbation of hamiltonian matrices. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 51(9):1755–1769, 2004.

[16] S. Grivet-Talocia. A perturbation scheme for passivity verification and enforcement

of parameterized macromodels. *IEEE Transactions on Components, Packaging and Manufacturing Technology*, 7(11):1869–1881, 2017.

[17] S. Grivet-Talocia, T. Bradde, M. De Stefano, and A. Zanco. A scalable reduced-order modeling algorithm for the construction of parameterized interconnect macromodels from scattering responses. In *IEEE Symposium on Electromagnetic Compatibility, Signal and Power Integrity*. IEEE, 2018.

[18] S. Grivet-Talocia and E. Fevola. Compact parameterized black-box modeling via fourier-rational approximations. *IEEE Transactions on Electromagnetic Compatibility*, 59(4):1133–1142, 2017.

[19] S. Grivet-Talocia and B. Gustavsen. *Passive macromodeling: Theory and applications*, volume 239. John Wiley & Sons, 2015.

[20] S. Grivet-Talocia, G. Signorini, S. B. Olivadese, C. Siviero, and P. Brenner. Thermal noise compliant synthesis of linear lumped macromodels. *Components, Packaging and Manufacturing Technology, IEEE Transactions on*, 5(1):75–85, Jan 2015.

[21] S. Grivet-Talocia and R. Trinchero. Behavioral, parameterized, and broadband modeling of wired interconnects with internal discontinuities. *IEEE Transactions on Electromagnetic Compatibility*, 60(1):77–85, 2018.

[22] N. J. Higham. *Accuracy and stability of numerical algorithms*. SIAM, Philadelphia, PA, 1996.

[23] T. Kailath. *Linear systems*. Prentice-Hall Englewood Cliffs, NJ, 1980.

[24] R. Kalman. On a new characterization of linear passive systems. 1964.

[25] A. C. S. Lima, B. Gustavsen, and A. B. Fernandes. Inaccuracies in network realization of rational models due to finite precision of RLC branches. In *International Conference on power System Transients (IPST)*, pages 1–5, 2007.

[26] L. Ljung. *System Identification: Theory for the User*. Prentice Hall, 1999.

[27] D. G. Luenberger. *Optimization by vector space methods*. Wiley-Interscience, 1997.

[28] A. F. Nikiforov, V. B. Uvarov, and R. P. Boas. *Special functions of mathematical physics*. Birkhäuser, 1988.

[29] S. B. Olivadese, G. Signorini, S. Grivet-Talocia, and P. Brenner. Parameterized and dc-compliant small-signal macromodels of rf circuit blocks. *IEEE Transactions on Components, Packaging and Manufacturing Technology*, 5(4):508–522, 2015.

[30] A. V. Oppenheim and A. S. Willsky. *Signals and systems*. Prentice Hall, Englewood Cliffs, NJ: Prentice Hall, 1983.

[31] J. Preibisch, T. Reuschel, K. Scharff, J. Balachandran, B. Sen, and C. Schuster. Exploring efficient variability-aware analysis method for high-speed digital link design using pce. *DesignCon, Jan*.

[32] C. Sanathanan and J. Koerner. Transfer function synthesis as a ratio of two complex polynomials. *Automatic Control, IEEE Transactions on*, 8(1):56–58, jan 1963.

[33] C. Sanathanan and J. Koerner. Transfer function synthesis as a ratio of two complex polynomials. *IEEE transactions on automatic control*, 8(1):56–58, 1963.

[34] C. Scherer and S. Weiland. Linear matrix inequalities in control. *Lecture Notes, Dutch Institute for Systems and Control, Delft, The Netherlands*, 3, 2000.

[35] P. Triverio, S. Grivet-Talocia, and M. Nakhla. An improved fitting algorithm for parametric macromodeling from tabulated data. In *Signal Propagation on Interconnects, 2008. SPI 2008. 12th IEEE Workshop on*, pages 1–4. IEEE, 2008.

[36] P. Triverio, S. Grivet-Talocia, and M. S. Nakhla. A parameterized macromodeling strategy with uniform stability test. *IEEE Transactions on Advanced Packaging*, 32(1):205–215, 2009.

[37] P. Triverio, M. Nakhla, and S. Grivet-Talocia. Parametric macromodeling of multiport networks from tabulated data. In *Electrical Performance of Electronic Packaging, 2007 IEEE*, pages 51–54. IEEE, 2007.

[38] P. Triverio, M. Nakhla, and S. Grivet-Talocia. Extraction of parametric circuit models from scattering parameters of passive rf components. In *Microwave Conference (EuMC), 2010 European*, pages 1635–1638. IEEE, 2010.

[39] P. Triverio, M. Nakhla, and S. Grivet-Talocia. Passive parametric macromodeling from sampled frequency data. In *Signal Propagation on Interconnects (SPI), 2010 IEEE 14th Workshop on*, pages 117–120. IEEE, 2010.

[40] C. F. Van Loan. Matrix computations (johns hopkins studies in mathematical sciences), 1996.

[41] M. Wohlers and E. Beltrami. Distribution theory as the basis of generalized passive-network analysis. *IEEE Transactions on Circuit Theory*, 12(2):164–170, 1965.

[42] M. R. Wohlers, N. Declaris, and H. G. Booker. Lumped and distributed passive networks: a generalized and advanced viewpoint. 1969.

[43] D. Youla, L. Castriota, and H. Carlin. Bounded real scattering matrices and the foundations of linear passive network theory. *IRE Transactions on Circuit Theory*, 6(1):102–124, 1959.

[44] A. Zanco. Adaptive algorithms for passivity verification and enforcement of multiparametric behavioral macromodels. Master's thesis, Politecnico di Torino, 2018.

[45] A. Zanco, S. Grivet-Talocia, T. Bradde, and M. De Stefano. Multivariate macromodeling with stability and passivity constraints. In *Signal Propagation on Interconnects, 2018. SPI 2018. 22th IEEE Workshop on*. IEEE, 2018.