GUSTAVO CORREIA NEVES CARVAS

Master of Science in Engineering and Management at Politecnico di Torino

SUSTAINABILITY OF ADDITIVE MANUFACTURING AND LCA: AN INTERACTIVE PROGRAM FOR COMPARING MANUFACTURING APPROACHES

Thesis for Master's Degree at Politecnico di Torino and Bachelor's Degree at POLI-USP.

Supervisors:

Prof. Fabrício Junqueira

Prof. Paolo C. Priarone

Torino

2023

To everyone who supported and accompanied me on this journey.

ACKNOWLEDGMENTS

I would like to take this opportunity to thank my family, for their continuous and amazing support for my education throughout my life, I would never have reached this moment without them and for that I will be forever thankful. My Mom, Elisa, my Dad, Claudio and my Brother, João Paulo I owe everything to you all, you've been an inspiration to me throughout my whole life, I love you all!

For my friends who have "fought" with me side by side in the trenches all these years, both at POLI-USP and Politecnico di Torino, without you the search for knowledge and betterment would have been dull and uninspiring. Special thanks to Eduardo Eiras, Luana Nunes and Rafael Sobral (the PEANOS), our assignments and study groups are some of the memories that I will cherish forever! Michele, I couldn't have asked for a better roommate and classmate in Torino!

Finally, many thanks to Marina Monteiro, who has been a huge support and a source of amazing energy for the past few years, I wouldn't be here without you.

"Climate change is the defining issue of our time – and we are at a defining moment."

– António Guterres, Secretary General of the United Nations

ABSTRACT

Additive Manufacturing has become more and more mainstream as technologies evolve and become more useful, although the perspective seems bright, it is undeniable that the technology has its limitations that inhibit the grater adoption of it nowadays. This thesis will seek to compile and discuss advantages and disadvantages of additive manufacturing as it pertains to sustainable manufacturing, namely Topological optimisation, Waste reduction, Low Scale Efficiency, Customization, Spare Parts, Reduction of inventory, Design Simplification, Time to market, Decentralized Manufacturing, Low Scalability, Productivity, High Embodied Impact Materials, High costs, Lack of experienced workforce, Adaptation of traditional designers, Information Asymmetry and Toxic Powder and Substances. Then the presence, impact and potential of AM in the Aerospace, Automotive and Medical industry will be discussed as well.

Additionally this thesis will present a decision making tool, for simulating LCAs by allowing the user to easily change initial inputs, through a user interface, in order to permit the visualization of positive impacts that can be captured through Additive Manufacturing as opposed to Traditional Manufacturing practices.

Keywords – Additive Manufacturing, Sustainability, Decision Making Tool, Sustainable Manufacturing, Life Cycle Assessment

MOTIVATION

During research and studies in both of my universities I came into contact with Additive Manufacturing Technologies, how they work, possible impacts and was really drawn into its possible impacts relating to sustainability. When given the opportunity to explore the subject itself for my thesis, researching the literature and developing the software to support a tool that will be able to facilitate tests and simulations to make better decisions, combining what I learned in my Bachelor's and my Master's I became really motivated to delve into the topic. The motivation for this thesis has been to participate and help with the advancement of our manufacturing processes in order to accelerate the change in manufacturing decision making.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AM - Additive Manufacturing

LCA - Life Cycle Assessment

TM - Traditional Manufacturing

CONTENTS

1 INTRODUCTION

Each and every day Global Warming is increasingly becoming a problem that cannot be ignored any longer. News of record high temperatures or never seen before storms are no longer rare, it's because of these consequences that Sustainability is gaining importance and relevance on people's minds and countries' policies. Of course this problem has many sides to it, one that has been present in public discourse for a a relatively long time is Sustainable Energy Sources, or Electric Cars. Both of these are seen as ways to hinder progress of global warming, but one facet that has not been given such importance or media attention until now is Sustainable Manufacturing.

As is defined later in section (1.1.4), sustainable manufacturing is a concept that is really relevant for the reality in which humanity is at. *"Climate change is the defining issue of our time – and we are at a defining moment."* - António Guterres, Secretary General of the United Nations, this quote is important as it truly emphasizes the point our society finds itself, a moment where action need to be taken in order to prevent the worst outcomes. There is a need to emphasize the positive impact that AM technology can have through a software that permits simulations and representations of those effects.

1.1 Definitions

1.1.1 Traditional Manufacturing

Traditional Manufacturing is the term used when referring to manufacturing processes that are normally used when producing an object. They encompass all machining, molding, forming, joining techniques.

1.1.2 Additive Manufacturing

It is the manufacturing process in which the desired three dimensional object is built layer by layer, until the complete object has been produced. A popular example of

Additive Manufacturing are the many 3D-printers widely available today for the public, however this manufacturing method encompasses a wide variety of technologies that can produce parts not only with plastic but with many other materials such as metals, concrete or even human skin.

1.1.3 Embodied Energy and Carbon Emission

It is the energy consumed per unit of mass, or carbon emitted per unit of mass, to produce a material from its ores and feedstock.

1.1.4 Sustainable Manufacturing

The concept of sustainable manufacturing is a complex one and one that still creates debate between scholars, the understanding that will be used in this thesis is fueled by the reflections provided in (ROSEN; KISHAWY, 2012).

In the past, and even today, manufacturing decision making was based solely on economic costs, as global warming and resource exhaustion was not a worry then. However as the years go by and consciousness of the problem grows, tools and practices to help decision makers address other areas of impact such as Social and Environmental have become more common, e.g. LCA, Carbon Footprint Estimation. Despite a common misconception, Sustainable Manufacturing is not about simply considering the environment when taking decisions, it is the concept of choosing and creating processes that take into account the three areas Economic, Environmental and Social, to create products in a way that minimises costs on all 3 pillars.

1.1.5 Life Cycle Assessment

Life Cycle Assessment serves the purpose of studying inputs and outputs of a product system, evaluating their consequences and finally interpreting the results of the study. It is defined in ISO14040 (International Standards Organization, 2006). It consists of 4 main steps:

1. Goal and scope definition
2. Inventory Analysis
3. Impact Assessment

4. Interpretation

One must define system boundaries when executing a LCA, the main decision is which phases to include, these are the existing phases present in a product's life cycle:

1. Material Production: Encompasses all the efforts that are made from extracting the raw material and processing it to be used for manufacturing;
2. Manufacturing: Encompasses the impacts caused by manufacturing of the part;
3. Transport: Encompasses the impacts of transporting the part from production to assembly or use;
4. Use: Encompasses the impacts that this part has during its normal use;
5. Disposal: Encompasses impacts generated by disposing of the part at the end of its life cycle

In this thesis's decision tool program we will be able to simulate three different boundaries:

1. Cradle to Grave:

   Simulation will consider every phase in the life cycle, from material production until the end of the product's life.

2. Cradle to Gate:

   Simulation will consider only material production and manufacturing phase, ignoring transport, use and disposal.

3. Gate to Gate:

   Simulation will consider only manufacturing phase, ignoring material production, transport, use and disposal

2 SUSTAINABILITY OF ADDITIVE MANUFACTURING

In this chapter the compilation of a literature review will be presented in order to discuss advantages and disadvantages of AM when compared to TM as it pertains to Sustainable Manufacturing. Afterwards the presence of AM in prominent industries will be discussed in order to exemplify its use nowadays.

2.1 Advantages

2.1.1 Topological optimisation/Light-weighing

One of the biggest direct advantages between AM and TM is Topological Optimisation. This tool in conjunction with material swaps allows the re-design of parts in order to utilize much less mass to achieve the same physical properties. This type of optimisation is not directly or intrinsically related to Additive Manufacturing, it could be done to certain parts even with Traditional Manufacturing but limitations are much stricter, (ZHU et al., 2021), as designs will have to be changed in order to accommodate these limitations, and parts will have to be segmented in order to enable their production and then further assembled, as a result the manufacturing process becomes more complicated and returns gained through the optimization are diminished, as can be seen in figure 3. AM however, enables designs with complex geometries that maximize efficiency and minimize mass, all in one whole part, making it so the greatest amount of value can be extracted from these operations.

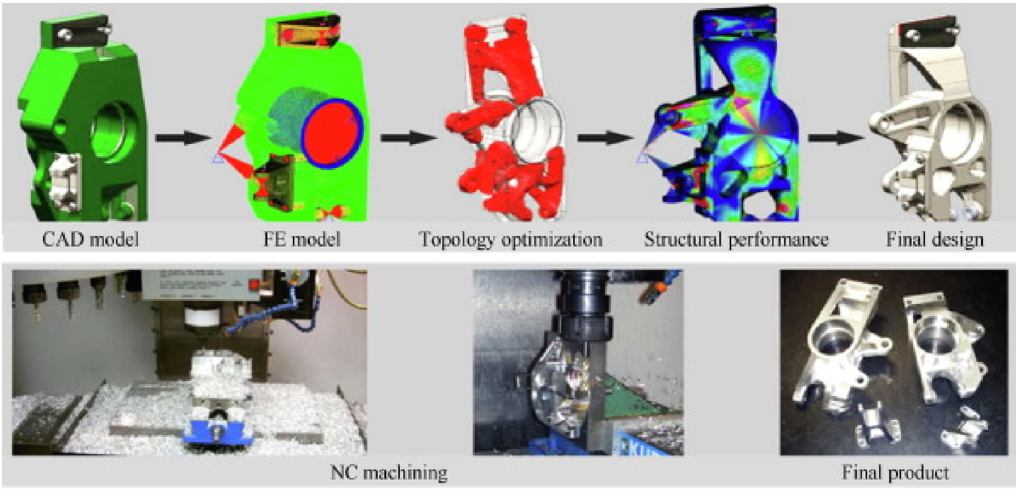


Figure 2: Example of topological optimisation in Traditional Manufacturing. (ZHU et al., 2021)

The process usually involves the following steps: inputting the original design CAD model into the optimisation program and feeding the program information about the application of the part: under which stresses will it be subjected to, from where and in which direction, relevant constraints and material that will be used. From there on out the program runs a series of FEA (Finite Element Analysis) to simulate the stresses on the part and understand which areas are most essential for the work. Then is starts removing mass through a variety of methods from specific areas and rerunning the simulations to check the stress response throughout the part, until it reaches the target that has been set(ABBEY, 2022). An important thing to note is that Topology Optimization is intrinsically related to the function the part will perform, if it were under different stresses and constraints the result would be wildly different, the process maximizes efficiency for the desired part in its desired context. An interesting synergy that emerges in AM is that many of digital design tools have direct connections and utilize the same file formats as optimization programs, creating an environment of compatibility which facilitates use and extraction of value from many tools quickly.

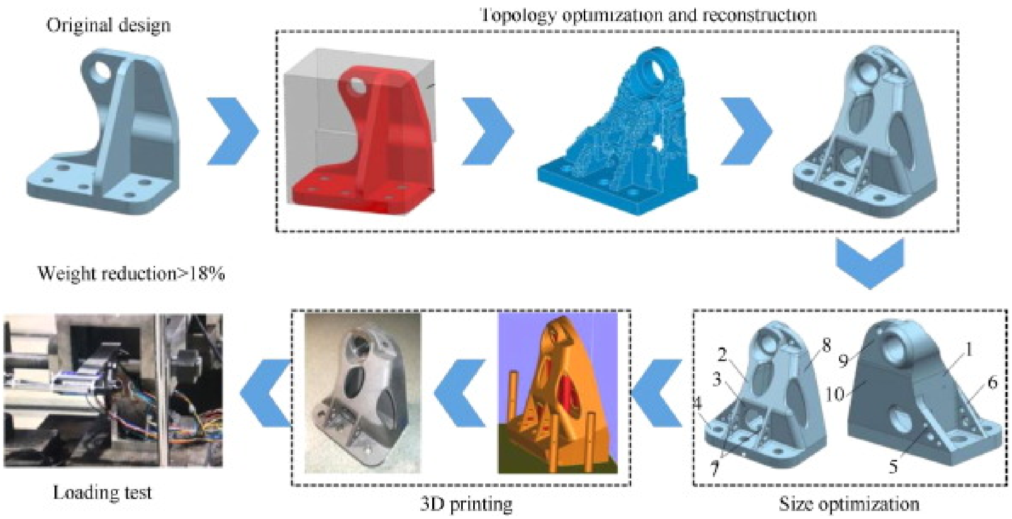


Figure 3: Example of topological optimisation in Additive Manufacturing. (ZHU et al., 2021)

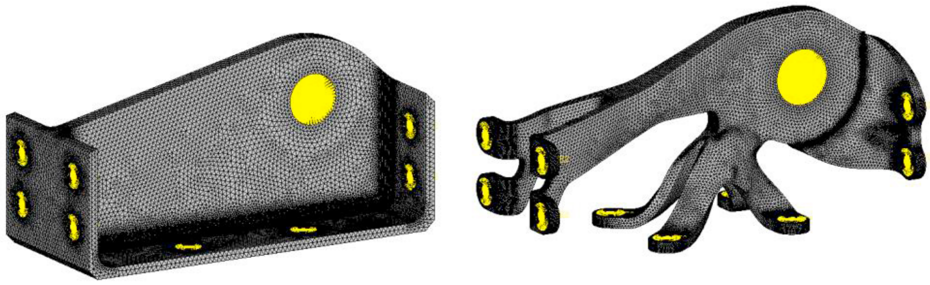


Figure 4: Mesh of an original component (left) and the optimized component (right) (SEABRA et al., 2016).

Through the lens of an LCA, the effects of this tool will be felt specially in the Manufacturing and Use Phase,(PRIARONE; CATALANO; SETTINERI, 2023) since this allows the use of much less mass in the product, in the former it means that less mass will have to be deposited, therefore leading to a smaller impact, and the latter is impacted because in many applications it will lead to reduced fuel consumption for example. This is the reason that has made AM parts become more frequent specially in industries where Fuel consumption is an important concern like in the Aeronautical Industry (GIBSON, 2017). An example can be seen in figure 5.

Figure 5: Part produced for use in Airbus A350XWB (EOS-GMBH,)

2.1.2 Waste reduction

By the simple nature of process with which AM works, less waste is produced. Whereas in TM in a machining process it's very likely one would begin with an initial workpiece that is much heavier than the final product, which then goes through a series of machining steps that removes material from the part until the desired geometry is achieved, AM creates the object layer by layer, using only the amount of material necessary to build the part, excluding a small amount of material that might be used for support. Consequently AM is much better suited to produce less waste than TM.

2.1.3 Low Scale Efficiency and Customization

An important advantage particular to AM is its low scale efficiency, i.e. its capacity to manufacture small numbers of products without the need of significant investment,(MANI; LYONS; GUPTA, 2014). This has to do with the fact that AM has a different cost structure when compared to TM, it does not require expenses in tooling, moulds, cutting implements or anything of the sort, the machine in AM virtually stays the same, the only necessary change being maybe the material that will be used. As such, the process achieves much better efficiencies when producing more unique parts.

One result of this advantage was captured early on in AM and utilized frequently by designers, which is AM as a great rapid prototyping tool. (GIBSON, 2017). Customization is another aspect which is becoming more and more important overtime. With Industry 4.0's trend of highly customized on-demand products, (PECH; VRCHOTA, 2022), AM fits perfectly into the necessity to produce these unique product cost effectively due to its flexibility and efficiency at low scale (ZAWADZKI; ŻYWICKI, 2016).

2.1.4 Spare Parts and Reduction of inventory

The adoption of AM also promotes advantages via a reduction of the general inventory level required to maintain operation, (GUTIERREZ-OSORIO et al., 2019). Such benefit is observed due to the facilitated process of on-demand replacement parts manufacturing, to the point of allowing parts to be printed even in remote locations, reducing the impact of the delivery of goods, and, therefore, shortening the supply chain, and largely reducing the need for stockpiling inventory. This results less costs being invested into inventory, as the need for area diminishes, driving rent and energy expenses down.

A practical example is the aerospace industry, in which AM shows the potential to largely reduce the need of part transportation into space, as part manufacturing in space is contemplated (ATTARAN, 2017).

2.1.5 Design Simplification and Time to market

Additive Manufacturing enables rapid prototyping, resulting in a reduced time to market and simplifying product design and development, (ATTARAN, 2017). Due to its nature, AM better accommodates changes in design than traditional manufacturing, resulting in a better stimulus to product design and innovation, as less penalties are associated to change as performing a large number of iterations results in minimal additional costs or losses of time. The perceived value of such advantage is even larger when considering that over 60% of designs submitted for tooling are modified while in production.

It is also worth mentioning the significant improvement in prototyping, granted by the design freedom of AM, which results in an overall reduction in the time-to-market. Finally, the high part specificity allowed by AM also contributes to the overall design processes, as compromises due to production feasibility are less susceptible (KELLENS et al., 2017), (ATTARAN, 2017).

Furthermore complex components normally necessitate an huge amount of assembling of simpler parts in TM, importantly there are negative consequences of this assembly such as lower reliability, greater inspection tooling and sustainment costs when compared to what would be experienced with a single part (GIBSON, 2017). AM with its flexibility and ability to deal with complexity without exploding costs enables feature integration, also known as part consolidation, further reducing design complexity and reverting the downsides of complex components produced with conventional means, as can be seen in figure 7.

2.1.6 Decentralized Manufacturing

TM manufacturing gains its biggest advantages by producing at large scale through centralized manufacturing facilities, that is factories that house many machines and are focused on producing the same part to be shipped out to other factories where they will be assembled into a complete product. AM flexibility and as was described in section 2.1.3 Low Scale Efficiency enable a different manufacturing organization. Decentralized manufacturing has many advantages and can outweigh the loss in economies of scale that would be incurred by switching from TM to AM (BAUMERS et al., 2016). Some direct consequences of decentralized AM are the following(ATTARAN, 2017):

1. Increased Agility and Responsiveness: Due to the characteristics of AM, a decentralized manufacturing structure would enable a company to produce parts on-demand locally, reducing lead times.

2. Reduced Logistic Costs: Due to the fact that production is made closer to the area in which it will be used, logistic costs are lowered.

3. Improved Product Customization: As production facilities are localized, companies are able to provide more personalized and customized products according to local demand.

4. Supply Chain Resilience: By spreading your production facilities in many locations, one is less prone to suffer supply chain disruptions due to problems in a single point of production.

2.2 Disadvantages

2.2.1 Low Scalability and productivity

While AM presents efficiencies in small scale productions, conversely it seems to falter when considering medium to large scale productions (ATTARAN, 2017). As already described previously on section 2.1.3, this can be justified by the different cost-structure between AM and TM (BAUMERS et al., 2016). TM experiences economies of scale from indivisibilities, as the costs of tooling, moulds and cutting implements are amortized over larger scales of production, driving unit cost down. However such expenses are not present in AM and this is therefore not experienced. Although there are indications that more productive platforms and higher build volumes are drivers that can be explored in order to drive unit cost down(BAUMERS et al., 2016).

Furthermore, production times are also very important aspect to consider, as nowadays production on AM technologies normally takes hours, where as the alternative can produce the same part in much less time, over larger numbers this becomes unsustainable for AM.

Still there is a possibility that these disadvantages might be offset by decentralized manufacturing (better described on section 2.1.6). Additionally another factor that likely will result in better unit cost is increasing Build Volume and maximizing utilisation of the volume will dilute costs such as warm-up or cool down (BAUMERS et al., 2014).

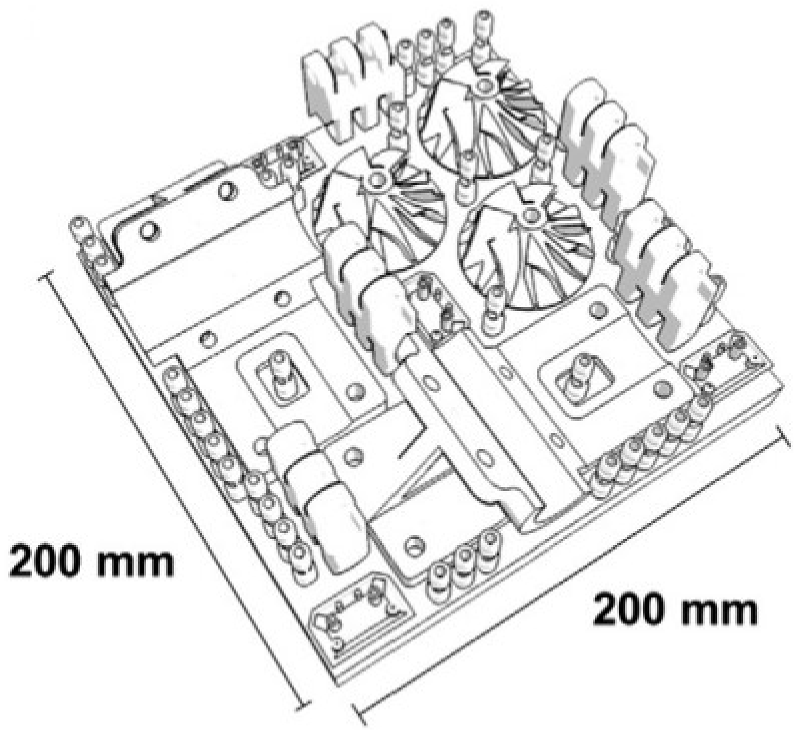


Figure 6: Example of maximum capacity utilization (BAUMERS et al., 2016)

2.2.2 High Embodied Energy/Carbon Materials

The materials used in AM present much greater embodied energy and carbon when compared to TM, this is due to two important facts necessary for AM material production:

- Atomization/Processing

  As an example let's consider an AM process in which a Aluminum part will be produced. Not only will there be the normal embodied impact of aluminum production, as there would be for an analogous machining process, but now the metal must go through an atomization process. This is normally done by gas atomization, where the metal is melted and then atomized by a strong flow of gas, commonly Argon, to create the powder that will be used for production. This increases further the embodied impact of this material.

- Use of higher quality materials

  Many times in order to achieve better Topological Optimization, a new material might be chosen that presents better properties, enabling the new design to be even slimmer. This is a trade-off between reduction in mass and the increase in embodied impact.

2.2.3 High costs

Though costs in general have been trending downwards for AM over recent years they are still a significant obstacle for the adoption of the upcoming technology. Costs are not only expressive for high grade materials and machines but also for maintenance, as a low amount of service providers and uncertainty in new AM technology makes it so maintenance prices are driven up (DWIVEDI; SRIVASTAVA; SRIVASTAVA, 2017).

This section has large overlap with what was explored in section 2.2.1 (Low Scalability and productivity) as these two remain the biggest barriers to more widespread AM adoption. Still the outlook is positive since as the technology develops and adoption becomes more common, more investments will be made and more companies will enter the market, driving costs further down and increasing the downward trend in prices in the near future.

2.2.4 Lack of experienced workforce

Even though Additive Manufacturing has its origins in the 1980s, it is only recently that commercial use of the technology has begun taking place. One of the consequences of this recent development is that the labour market is not yet prepared to fill the roles required, in other words, there is a limited pool of professionals with the necessary experience and training throughout the whole value chain (DWIVEDI; SRIVASTAVA; SRIVASTAVA, 2017).

A factor that continues to complicate this situation is the rapidly evolving nature of the technology. With new developments each day it is hard to keep up with the most up to date training and techniques. Of course this creates problems for companies looking to focus and rely on this technology as in order to stay competitive a reasonable and continuous effort must be made (VITALE; COTTELEER; HOLDOWSKY, 2016) (DESPEISSE; MINSHALL, 2017). This struggle is accompanied by a still underrepresented number of education programs focused on exploring these new technologies. As the field expands and adoption becomes more widespread, it is expected for this demand for skilled workers to only grow.

2.2.5 Adaptation of traditional designers

Building upon the last section's (2.2.4) points, there is an important barrier present in the adaptation of traditional designers to a new AM paradigm, as they have gained experiences and practices related to TM. Additive Manufacturing presents unique challenges and possibilities, ones that traditional designers will not be accustomed and open to. At the same time that AM provides a whole new range of possibilities with greater flexibility and ability to customize parts, greater geometrical freedom, topological optimization, it also comes with limitations such as support structures, how to design overhangs and so forth (DWIVEDI; SRIVASTAVA; SRIVASTAVA, 2017).

Adoption of AM requires the adaptation and creation of new practices in line with the realities of the new technology (MELLOR; HAO; ZHANG, 2014). This barrier can be an important one for the widespread adoption of AM and surpassing it will take great investment and effort in training, tools and processes. The same could also be expanded to a barrier related by Workers' resistances to switching to this new technology (DWIVEDI; SRIVASTAVA; SRIVASTAVA, 2017).

2.2.6 Information Asymmetry

This concept describes a situation in which there is an unequal distribution of information between different parties, which can then lead to misunderstandings and inefficiencies. This term is relevant to AM since as has been discussed in sections 2.2.4 and 2.2.5, there is a lack of general knowledge about Additive Manufacturing. A company might not be aware of the capabilities and limitations of the technology, the lack of readily available and distributed knowledge makes it so companies are more conservative and afraid on whether or not to begin using AM (DWIVEDI; SRIVASTAVA; SRIVASTAVA, 2017).

The lack of standardized protocols, guidelines and standards also creates uncertainties

(MELLOR; HAO; ZHANG, 2014) as companies are unfamiliar with the technology and might be cynical when it comes to the quality of the product they are buying, slowing its adoption. It is essential for standards to be set and for efforts to close the information asymetry in order to clear up misunderstandings. This way AM can become more efficient and more accepted.

2.2.7 Toxic Powder and Substances

There are health hazards to consider when discussing AM. They are mainly related to chemicals/solvents used in support removal which can cause skin reactions, eye irritation, and allergies (ABDULHAMEED et al., 2019).

Another point of attention relates to the handling and contact with the various metal powders present in AM, not much literature has been produced on the harmfulness and toxicity of these materials, although emissions of ultra-fine particles have been reported and might be a cause for concern, this must be investigated and understood further for a safer adoption in the future. As for now the new technology does not require the use of a lot of common harmful chemicals present in TM such as cutting fluids and lubricants which already presents a favorable scenario for the future (PENG et al., 2018).

2.3 Sustainability Table

Now that the advantages and disadvantages have been presented, it might be useful to position them on atable, categorizing them by which of the pillars of Sustainable Manufacturing they impact.

Advantages	**Economic**	**Social**	**Environment**
Topological optimisation	✓		✓
Waste reduction	✓		✓
Low Scale Efficiency and Customization	✓	✓	✓
Spare Parts and Reduction of inventory	✓		✓
Design Simplification and Time to market	✓		
Decentralized Manufacturing	✓	✓	
Disadvantages			
Low Scalability and productivity	X		
High Embodied Energy/ Carbon Materials			X
High costs	X		
Lack of experienced workforce	X	X	
Adaptation of traditional designers	X	X	
Information Asymmetry	X		
Toxic Powder and Substances		X	

Table 1: Summary of Advantages and Disadvantages distribution among the three Sustainable Development areas.

2.4 Use in Industry

Finally to better illustrate the effects of these advantages and disadvantages related to AM, in this section uses in 3 industries will be presented.

2.4.1 Aerospace

The Aerospace industry might be the industry where AM has experienced its greatest adoption. This is due to the high performance that is required in this sector, leading to development and manufacturing costs to be less relevant than improvements in weight reduction and fuel consumption (GIBSON, 2017).

Also relatively speaking the aerospace industry has a much smaller scale when compared to the automotive industry for example, meaning that the disadvantages of large scale production are less felt, but improvements in design freedom and topological optimization can be fully explored and profited from. As was described in section 2.1.5 part consolidation also serves as a great benefit for this industry, improving performance and reliability. As many aircraft have long lifespans, much of the industry revolves around maintaining and replacing parts, which aligns very well with AM ability to produce on demand customized parts, also reducing inventory costs, as was discussed in section 2.1.4 (NAJMON; RAEISI; TOVAR, 2019).

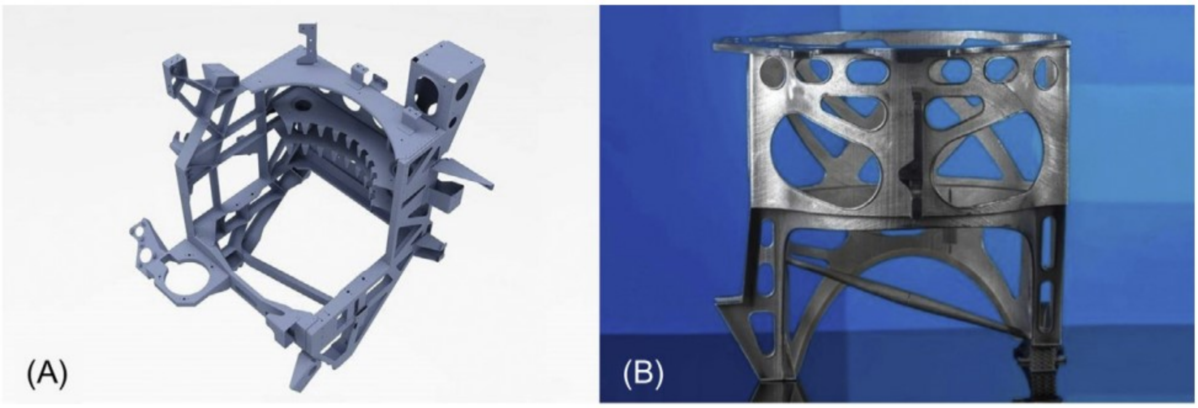


Figure 7: Example of part consolidation. AM hydraulic reservoir from Airbus consolidating 126 parts (NAJMON; RAEISI; TOVAR, 2019).

The reasons listed are why Boeing and Airbus, leaders of the aerospace industry, have adopted AM as an essential part of their manufacturing process.

2.4.2 Automotive

Similar to the aerospace industry, the automotive industry places great relevance into gains in performance and fuel consumption, however the relationship between cost and performance is different than the aeronautical industry. Combined with the fact that the scale of production is much higher in comparison, there are more barriers present here than in aerospace. This does not mean that AM is not present or utilized, only that it has to overcome bigger barriers in order to do so. As the technology advances and evolves there is great potential for very positive impact in this industry with more widespread AM presence.

As it stands the further adoption of this technology is hindered by its difficulty at operating at higher productivity levels (DWIVEDI; SRIVASTAVA; SRIVASTAVA, 2017) in order to keep up with industry demands.

2.4.3 Medical

The relevance of AM in the medical industry is directly related with is great customization and on-demand capacities. Through them, the industry can create patient specific solutions like customized implants, surgical guides, prosthetics and orthotics, seen in figure 8 for example. They allow for much faster recoveries. In conjunction with recent advancements in technologies that enable acquiring high quality 3D scans of the human body more and more have these solutions gained importance (GIBSON, 2017), (JAVAID; HALEEM, 2018).

Figure 8: Example of orthotic developed using AM technology (PETLOCK. . . ,)

Very important positive impact of AM in the medical industry was shown very recently throughout the COVID-19 Pandemic as the technology was used to supplement the supply chain with 3D face masks, face shields, hand gloves and parts of medical ventilators (GARCIA et al., 2021).

3 A SOFTWARE FOR SUPPORTING DECISION MAKING

In this chapter the developed software will be explained and discussed. First the scientific references of the model and the model itself, then system boundaries used in the simulation. After these initial clarifications, the program itself and its details: Libraries, variables and functions and finally 3 simulations.

3.1 Model

The model used for the development has been extended from many articles and authors available in the literature such for example the following: (PRIARONE et al., 2016) (KARA; LI, 2011), and it has proved to be effective in analyzing such types of problem and was therefore chosen for this software.

It is a model that considers series of stages in the life cycle of a part and calculates the energy demand and carbon impact in each by considering the material inputs and outputs each stage requires. The exact boundaries and considerations can be changed depending on the objective of the study, for example whether or not to consider the impact of cutting fluids on a milling process or to whether or not consider the impact of material production or disposal.

Typically the phases that are considered are the following:

1. **Material Production:** This phase denotes the phase in which the raw material is collected and processed until it becomes ready to serve as input for Manufacturing.

2. **Manufacturing:** This phase denotes the step in which the raw material will be turned into the desired part by one or more manufacturing steps.

3. **Transportation:** This phase denotes the transportation of the part from where it was manufactured to where it will be used/assembled into the complete product (such as an airplane)

4. **Use:** This phase encompasses the time in which the part will put in use

5. **Disposal:** Finally, this phase denotes the way in which the part will be discarded, whether it will be recycled, go into a landfill or any other form of disposal.

For each of these steps a formula used to calculate the impacts generated for each step due to material inputs and outputs. They will be further explained and described on section 3.3.3. Also the flow of the model can be seen on figure 9.

Regarding the eco-data that will be used as input, it can be hard to come by and find trustworthy information. Furthermore they are also very dependent on where the analysis is taking place, for example the embodied energy of aluminum has very large variance relating to many factors (INGARAO et al., 2018). In the simulations presented in this paper, inputs were sourced from available literature, such as AM processes energy demand (KELLENS et al., 2017), Use phase gains (PRIARONE; INGARAO, 2017), embodied energy (PRIARONE et al., 2016) and (PRIARONE; CATALANO; SETTINERI, 2023) and others. Scenario 1 has some data that was not sourced from literature since its purpose is to simulate how the betterment and development of the technology in the next years might change the scenario observed in the validation scenario.

3.2 Simulation boundaries and Material Flows

Simulation boundaries for the LCA program will be defined by the user. The options available are as follows:

1. Cradle to Grave:

   Simulation will consider every phase in the life cycle, from material production until the end of the product's life.

2. Cradle to Gate:

   Simulation will consider only material production and manufacturing phase, ignoring transport, use and disposal.

3. Gate to Gate:

   Simulation will consider only manufacturing phase, ignoring material production, transport, use and disposal

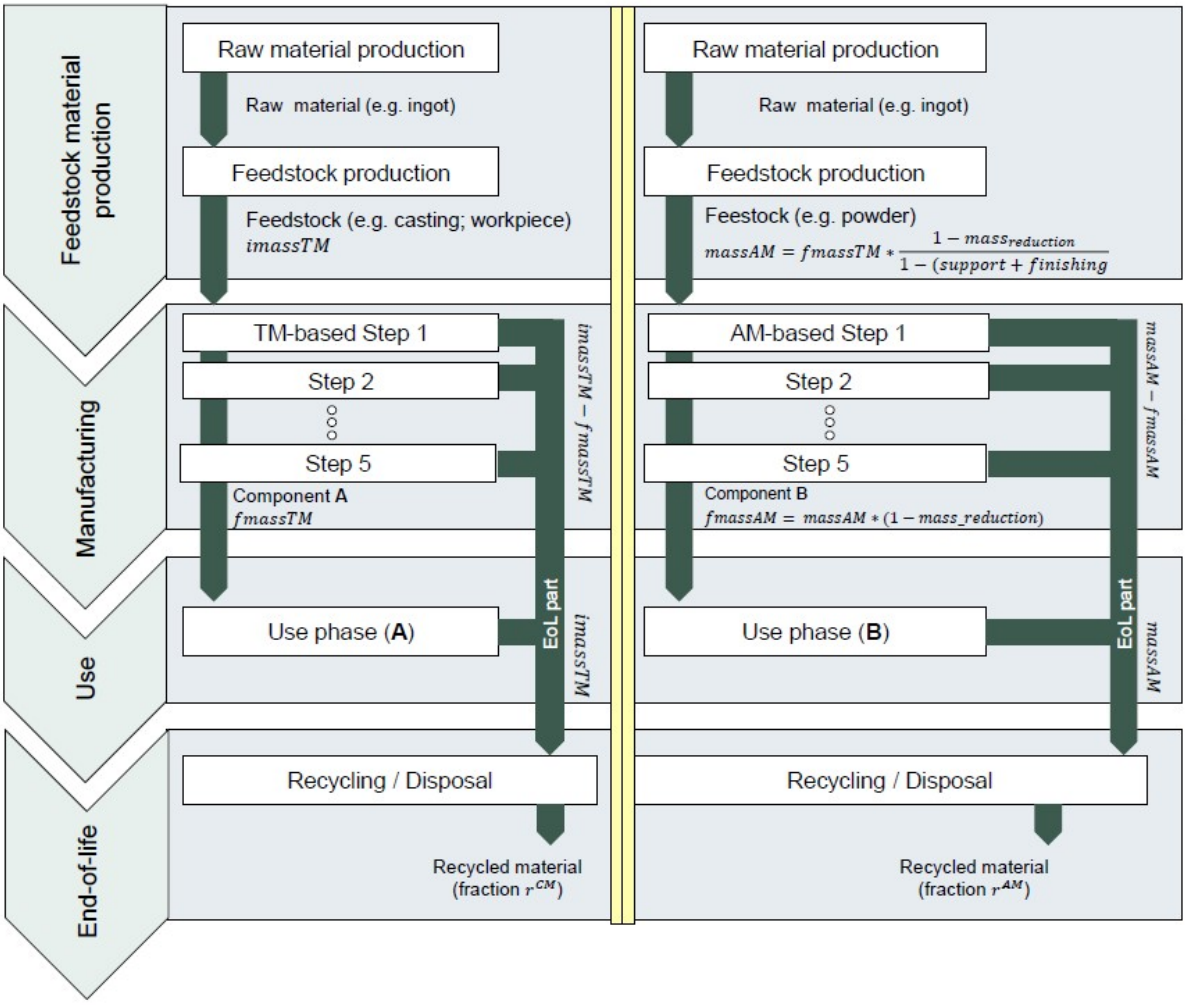


Figure 9: Diagram of Material Flows in Cradle to Grave Assessment, extended from (PRIARONE; CATALANO; SETTINERI, 2023)

3.3 Program

In this section the program's details will be explained and presented.

The program was developed in Python 3.8.3 and specifically using a Jupyter Notebook environment. It's available for download here: ⟨https://github.com/guscarvas/LCA⟩

3.3.1 Libraries

NumPy

This is a very commonly used library in Python, on it's own website the library is defined as "is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.". In this program the library is mainly used for its multiplication functions for arrays for better performance.

(WHAT...,)

Matplotlib

This library is one that is also commonly used and one that anyone that learns python should be familiar with. Matplotlib is a 2D plotting library and it enables the plotting of a wide range of graphs, E.g line plots, bar plots, scatter plots, pie charts and many more. It is designed to be very customizable by providing many commands and options to change details about the plots.

Jupyter Widgets - ipywidgets

This library is at the core of the interactivity of this program, without it the user would have to run the program again manually everytime.

Jupyter Widgets is a library of interactive widgets for Jupyter Notebooks, as in it provides an API for creating UI controls for inteactivity, the ones that are used in the program, like text inputs, sliders and buttons. The library therefore allows the creation of interactive applications, and even dashboards.

3.3.2 Input Variables

They are the variables that the user will have direct control over, able to change their value through the user interface that is provided. In order to facilitate the use in the program these variables and better understand their effect, they are explained through 5 bullets: Description, Varibale Type, "Can be set to" (describes what values the variable is designed to support), Input (How the user may interact and change the variable's value) and Functions in which it is present, they will be described afterwards.

- Mass
  - <u>imassTM</u>
    - **Description:** Stores the mass (kg) of the initial workpiece to be subtracted with TM processes;

    * **Variable type:** Float;
    * **Can be set to:** Any positive value;
    * **Input:** Through slider;
  - fmassTM
    * **Description:** Stores the mass (kg) of the finished workpiece produced with TM;
    * **Variable type:** Float;
    * **Can be set to:** Any positive value;
    * **Input:** Through slider;

- Material properties
  - Virgin Material Embodied Impact
    * embTMe
      · **Description:** Embodied Energy (MJ/kg) of the material used in the TM path;
      · **Variable type:** Float;
      · **Can be set to:** Any positive value;
      · **Input:** Through text input;
    * embAMe
      · **Description:** Embodied Energy (MJ/kg) of the material used in the AM path;
      · **Variable type:** Float;
      · **Can be set to:** Any positive value;
      · **Input:** Through text input;
    * embTMc
      · **Description:** Embodied Carbon (kgCO_2/kg) of the material used in the TM path;
      · **Variable type:** Float;
      · **Can be set to:** Any positive value;
      · **Input:** Through text input;
    * embAMc
      · **Description:** Embodied Carbon (kgCO_2/kg) of the material used in the AM path;

· **Variable type:** Float;

· **Can be set to:** Any positive value;

· **Input:** Through text input;

– Disposal Embodied Impact

∗ embEDisTM

· **Description:** Stores the embodied energy (MJ/kg) for disposal of material per unit of material used in TM;

· **Variable type:** Float;

· **Can be set to:** Any positive value;

· **Input:** Through text input;

∗ embCDisTM

· **Description:** Stores the embodied carbon (kgCO_2/kg) for disposal of material per unit of material used in TM;

· **Variable type:** Float;

· **Can be set to:** Any positive value;

· **Input:** Through text input;

∗ embEDisAM

· **Description:** Stores the embodied energy (MJ/kg) for disposal of material per unit of material used in AM;

· **Variable type:** Float;

· **Can be set to:** Any positive value;

· **Input:** Through text input;

∗ embCDisAM

· **Description:** Stores the embodied carbon (kgCO_2/kg) for disposal of material per unit of material used in AM;

· **Variable type:** Float;

· **Can be set to:** Any positive value;

· **Input:** Through text input;

– Recycled Embodied Impact

∗ embERecTM

· **Description:** Stores the embodied energy impact (MJ/kg) from recycled material input per unit of material used in TM;

        - **Variable type:** Float;
        - **Can be set to:** Any positive value;
        - **Input:** Through text input;
    - embCRecTM
        - **Description:** Stores the embodied carbon (kgCO_2/kg) from recycled material input per unit of material used in TM;
        - **Variable type:** Float;
        - **Can be set to:** Any positive value;
        - **Input:** Through text input;
    - embERecAM
        - **Description:** Stores the embodied energy impact (MJ/kg) from recycled material input per unit of material used in AM;
        - **Variable type:** Float;
        - **Can be set to:** Any positive value;
        - **Input:** Through text input;
    - embCRecAM
        - **Description:** Stores the embodied carbon (kgCO_2/kg) from recycled material input per unit of material used in AM;
        - **Variable type:** Float;
        - **Can be set to:** Any positive value;
        - **Input:** Through text input;

- Manufacturing
  - Traditional Manufacturing
    - TM"X"1
        - **Description:** Stores the energy usage per unit of mass removed/heated/... (MJ/kg) in this Xth step. X can be substituted by 1,2,3,4 or 5, it point out which step is being analyzed at the moment in the TM path;
        - **Variable type:** Float;
        - **Can be set to:** Any positive value;
        - **Input:** Through text input;
    - TM"X"2

        - **Description:** Stores the CO_2 equivalent emitted per unit of mass removed/heated/... (kgCO_2/kg) in this Xth step. X can be substituted by 1,2,3,4 or 5, it point out which step is being analyzed at the moment in the TM path;
        - **Variable type:** Float;
        - **Can be set to:** Any positive value;
        - **Input:** Through text input;
    - <u>TM"X"3</u>
        - **Description:** Stores the percentage (%) of total mass that will be subtracted/used in this Xth step, e.g. If it is a machining step and 10% of mass will be removed, TMX3 will be 0.1 for this Xth step. X can be substituted by 1,2,3,4 or 5, it point out which step is being analyzed at the moment in the TM path;
        - **Variable type:** Float;
        - **Can be set to:** Any value from 0 to 1;
        - **Input:** Through slider;
    - <u>TMsteps</u>
        - **Description:** Stores the number of steps that will be considered on this simulation of the TM path, E.g. if you are considering a path where there will be first a milling step and then a finishing step, you will set this number to 2;
        - **Variable type:** integer;
        - **Can be set to:** Integer from 1 to 5;
        - **Input:** Through slider;
- Additive Manufacturing
    - <u>AM"X"1</u>
        - **Description:** Stores the energy usage per unit of mass placed/removed (MJ/kg) in this Xth step. X can be substituted by 1,2,3,4 or 5, it point out which step is being analyzed at the moment in the AM path;
        - **Variable type:** Float;
        - **Can be set to:** Any positive value;
        - **Input:** Through text input;
    - <u>AM"X"2</u>

- **Description:** Stores the CO_2 equivalent emitted per unit of mass placed/removed (kgCO_2/kg) in this Xth step. X can be substituted by 1,2,3,4 or 5, it point out which step is being analyzed at the moment in the AM path;
- **Variable type:** Float;
- **Can be set to:** Any positive value;
- **Input:** Through text input;

* AM"X"3
  - **Description:** Stores the percentage (%) of the total mass that will be placed/removed in this Xth step. X can be substituted by 1,2,3,4 or 5, it point out which step is being analyzed at the moment in the AM path;
  - **Variable type:** Float;
  - **Can be set to:** Any value from 0 to 1;
  - **Input:** Through slider;
* AMsteps
  - **Description:** Stores the number of steps that will be considered on this simulation of the AM path, E.g. if you are considering a path where there will be first a Stereolithography step and then a finishing step, you will set this number to 2;
  - **Variable type:** integer;
  - **Can be set to:** Integer from 1 to 5;
  - **Input:** Through slider;

- Percentages
  - mass_reduc_perc
    * **Description:** Stores the percentage of mass that will be reduced due to optimization of the part by producing it using AM technology;
    * **Variable type:** Float;
    * **Can be set to:** Any value from 0 to 1;
    * **Input:** Through a slider;
  - finishing_perc
    * **Description:** Stores the percentage of mass of the part that will need to be removed in a Finishing step to complete it;

    * **Variable type:** Float;
    * **Can be set to:** Any value from 0 to 1;
    * **Input:** Through a slider;
  - support_perc
    * **Description:** Stores the percentage of additional mass that is used to create supports by the AM technology when producing the desired part;
    * **Variable type:** Float;
    * **Can be set to:** Any value from 0 to 1;
    * **Input:** Through a slider;
  - recTM
    * **Description:** Stores the percentage of TM's material that can be recycled after its life cycle is done, used in the Substitution method for evaluating the impact of recycling;
    * **Variable type:** Float;
    * **Can be set to:** Any value from 0 to 1;
    * **Input:** Through a slider;
  - recAM
    * **Description:** Stores the percentage of AM's material that can be recycled after its life cycle is done, used in the Substitution method for evaluating the impact of recycling;
    * **Variable type:** Float;
    * **Can be set to:** Any value from 0 to 1;
    * **Input:** Through a slider;
- Power Grid
  - eff_energ_conv
    * **Description:** Stores the energy conversion efficiency of the local Power Grid that is being used for manufacturing;
    * **Variable type:** Float;
    * **Can be set to:** Any value from 0 to 1;
    * **Input:** Through a slider;
  - CO2equivalent

    * **Description:** Stores the equivalent of CO_2 that is emitted by the unit of energy used (kgCO_2/MJ) during the manufacturing phase;
    * **Variable type:** Float;
    * **Can be set to:** Any positive value;
    * **Input:** Through text input;

- Use Type
  - useType
    * **Description:** Stores the input of where the user wants to consider the part to be used in;
    * **Variable type:** Float;
    * **Can be set to:**
      · *All* : Display energy and CO_2 graphs for both TM and AM;
      · *Difference* : Displays a energy and a CO_2 graph with the difference for each phase of the product life cycle ;
    * **Input:** Through drop down menu;
- Transport
  - Method
    * transpTM
      · **Description:** Stores the name of the transportation mode that will be used for TM, serves as a key for the dictionary of specific impacts for each transportation method;
      · **Variable type:** string;
      · **Can be set to:** 17 different values
      · **Input:** Through dropdown menu;
    * transpAM
      · **Description:** Stores the name of the transportation mode that will be used for AM, serves as a key for the dictionary of specific impacts for each transportation method;
      · **Variable type:** string;
      · **Can be set to:** 17 different values
      · **Input:** Through dropdown menu;

- Distances
  - <u>distTM</u>
    - **Description:** Stores the distance that will be travelled between manufacturing and Use phase for the TM path;
    - **Variable type:** Float;
    - **Can be set to:** 17 different values
    - **Input:** Through dropdown menu;
  - <u>distAM</u>
    - **Description:** Stores the distance that will be travelled between manufacturing and Use phase for the TM path;
    - **Variable type:** Float;
    - **Can be set to:** 17 different values
    - **Input:** Through dropdown menu;

- Control
  - <u>solver</u>
    - **Description:** Serves as a flag from the user to indicate whether the breakeven percentage of mass reduction from TM to AM should be calculated;
    - **Variable type:** Boolean;
    - **Can be set to:** True or False
    - **Input:** Through checkbox;
  - <u>gType</u>
    - **Description:** Serves as an input from the user to decide the type of graph they want to see;
    - **Variable type:** String;
    - **Can be set to:**
      - *All* : Display energy and CO_2 graphs for both TM and AM;
      - *Difference* : Displays a energy and a CO_2 graph with the difference for each phase of the product life cycle ;
    - **Input:** Through drop down menu;
  - <u>secondary</u>

    - **Description:** Serves as flag to indicate if manufacturing inputs are as primary or secondary energy;
    - **Variable type:** Boolean;
    - **Can be set to:** True or False
    - **Input:** Through checkbox;
  - boundary
    - **Description:** Serves ;
    - **Variable type:** Boolean;
    - **Can be set to:**
      - *Cradle to Grave* : Display energy and CO_2 graphs for both TM and AM;
      - *Cradle to Gate* : Displays a energy and a CO_2 graph with the difference for each phase of the product life cycle ;
      - *Gate to Gate* : Displays a energy and a CO_2 graph with the difference for each phase of the product life cycle ;
    - **Input:** Through dropdown menu;

3.3.3 Functions

In this subsection all the functions present in the program will be described.

Material Production

The formula displayed on equation 3.1 is used to calculate the material production impact of the material that will be used in manufacturing.

$$E_{MP} = embodied_{impact} * mass \tag{3.1}$$

- E_{MP} : Total energy demand/carbon emission in material production phase (MJ) or $(kgCO_2)$;
- $embodied_{impact}$: Embodied impacts for the material that will be used in manufacturing;
- mass: Mass of the input material produced for the manufacturing phase;

embodied energy is meant to represent the energy required to extract raw materials and process them before they are ready for use in the manufacturing phase. They can vary wildly even when considereing the same material, for example Aluminum, which has the embodied energy for primary production of 204MJ/kg wih a standard deviation equals to 51MJ/kg (INGARAO et al., 2018)

This function will calculate material production energy requirements and CO2 emissions for both TM and AM simultaneously by multiplying the array of embodied energy and carbon by the mass relevant for the given path. This variability can come from many factors such as difference in energy-mix between countries, location of where the aluminum is being extracted, difficulty in setting system boundaries or any other problem that might complicate the definition of eco-attributes.

- Variables
  - EmbodiedMat1
    - **Description:** Array with the embodied energy (MJ/kg) and carbon (kgCO2/kg) of the material used in TM, - [Energy, Carbon];
    - **Variable type:** Array;
  - EmbodiedMat2
    - **Description:** Array with the embodied energy (MJ/kg) and carbon (kgCO2/kg) of the material used in AM, - [Energy, Carbon];
    - **Variable type:** Array;
  - massTM
    - **Description:** Stores the mass of the initial workpiece that will be used in TM path;
    - **Variable type:** Array;
  - massAM
    - **Description:** Stores the mass of the mass that will be placed on the AM path (has to consider support mass and allowance of the machine);
    - **Variable type:** Array;
- Returns
  - EG_TM
    - **Description:** Stores the energy and carbon demand for the material production for the workpiece in TM;

    - **Variable type:** Array;

  - EG_AM

    - **Description:** Stores the energy and carbon demand for the material production of the material that will be placed in AM;
    - **Variable type:** Array;

Manufacturing

$$E_M = SEC_1 * m_1 + SEC_5 * m_5 + ... + SEC_5 * m_5 \tag{3.2}$$

- E_M: Total energy demand/carbon emissions for manufacturing phase (MJ) or $(kgCO_2)$;
- SEC_x: Specific Energy Consumption for x step in the manufacturing phase (MJ/kg) or $(kgCO_2/kg)$;
- m_x: mass that will be removed/place/heated in this x step (kg);

This function will calculate the energy and carbon cost of manufacturing in each of the manufacturing paths by multiplying the Energy demand (MJ/kg) and carbon emission (kgCO2/kg) for each step by the mass that will be removed/placed in each step.

- Variables

  - processing_steps

    - **Description:** Each of the rows in the array is meant to represent one step in the manufacturing process, for example, if you have a milling step then a finishing step, the first row will relate to the costs and mass removed of the milling step and the finishing step will be described on the second row - each of the 5 rows is in this format:
      [Specific Energy Demand, Specific CO2 Emission, Mass];
    - **Variable type:** Array;

- Returns

  - result

  - **Description:** Stores the result of the multiplication and sum of specific impacts for the manufacturing steps. This is its format:
    ["Total Energy demand", "Total $kgCO_2$ emitted"];
  - **Variable type:** Array;

Transport

$$E_T = d * m_f * STE \tag{3.3}$$

- E_T: Energy demand/Carbon Footprint of transport phase (MJ) or ($kgCO_2$);
- m_f: final mass of the product to be transported (kg);
- d: distance that will be covered in transport (km);
- STE: Specific Transport Energy/Carbon, ($MJ/(kg * m)$) or ($kgCO_2/(kg * m)$);

Values used in the program for each transportation method were gathered by (ASHBY, 2012)

- Variables
  - mass
    - **Description:** Simply stores the mass of the product that will be transported;
    - **Variable type:** Float;
  - distance
    - **Description:** Stores the method of transportation, which then will be used as a key for a dictionary in the function to gather the transportation impacts per metric ton * km;
    - **Variable type:** Float;
  - method
    - **Description:** Stores the method of transportation, which then will be used as a key for a dictionary in the function to gather the transportation impacts per metric ton * km;

        * **Variable type:** String;

- Returns

    - result

        * **Description:** Stores the result of the multiplication of mass, distance and specific transport impacts in this format:
          [Total Energy Demand, Total CO_2] emission;
        * **Variable type:** Array;

Use

Instead of calculating impacts for both AM and TM, this phase only the gains that AM enables will be calculated, therefore it will be the difference between the Use phase of TM and AM.

$$E_U = m_{reduction} * SUPG \tag{3.4}$$

- E_U: Total energy demand/carbon demand reduction due to light-weighing in Use phase (MJ) or ($kgCO_2$);
- $m_{reduction}$: mass reduction achieved in light-weighing (kg);
- SUPG: Specific Use Phase Gains, represents the savings per kg of reduces mass (MJ/kg) or ($kgCO_2$/kg);

This function will calculate the "gain" (the decrease of environmental cost due to bigger efficiencies in geometry in AM) in energy demand and carbon emissions when the produced part is in use. (PRIARONE; INGARAO, 2017)

- Variables

    - useType

        * **Description:** Will be used as a key to the dictionaries of energy savings and carbon demand decrease per kg of decreased weight;
        * **Variable type:** String;

    - dif_mass

    * **Description:** Stores the difference in mass between the mass of the finalized product manufactured through TM and AM;
    * **Variable type:** Float;

- Returns
  - result
    * **Description:** Stores the multiplication of the energy and a carbon gains in an array in this format:
      [Energy Savings, CO2 savings];
    * **Variable type:** Array;

Disposal

The *Substitution Method* will be used as the method of benefit awarding impact of recycling. It is calculated like so:

$$E_D = r * e_r + (1 - r) * (e_v + e_d) \tag{3.5}$$

- E_D: Total energy demand or carbon emission for the disposal phase (MJ) or ($kgCO_2$);
- r: recyclability of the material;
- e_r: The embodied impact for recycled of material per unit of material (MJ/kg) or ($kgCO_2$/kg);
- e_v: The embodied impact for virgin material per unit of material (MJ/kg) or ($kgCO_2$/kg);
- e_d: The embodied impact for disposal of material per unit of material (MJ/kg) or ($kgCO_2$/kg);

- Variables
  - recyclability
    * **Description:** % of recyclability (combined recovery, recycling rate and recycling mateiral yield);
    * **Variable type:** Float;

- emb_virgin
  - **Description:** Stores the embodied impact from virgin material input per unit of material;
  - **Variable type:** Float;
- emb_waste
  - **Description:** Stores the embodied impact for disposal of material per unit of material;
  - **Variable type:** Float;
- emb_rec
  - **Description:** Stores the embodied impact from recycled material input per unit of material;
  - **Variable type:** Float;
- mass
  - **Description:** Stores the mass that is being disposed of;
  - **Variable type:** Float;

- Returns
  - result
    - **Description:** Stores the result of the substitution method in this format: [Energy demand, CO_2 emission];
    - **Variable type:** Array;

Calculation

The inputs to this function are encompassed the same as the ones presented in section 3.3.2 and therefore will not be repeated here.

This is the core function of the program as it pertains to the calculation, it controls and calls all the other functions into action. It has been separated from the main function since it is called more than once, and in order to be more efficient in code it was separated

Read Input

This is a quality of life function that enables the user to read inputs that is present on the simulation at the moment

- Variables
  - file_name
    - **Description:** Stores the name of the file from which the inputs will be extracted;
    - **Variable type:** String;
- Returns
  - inputDict
    - **Description:** Dictionary of inputs, each key is the name of the variable and the values are the input values;
    - **Variable type:** Dictionary;

Completed

The inputs to this function are encompassed the same as the ones presented in section 3.3.2 and therefore will not be repeated here.

This function calls the Calculation function and is called by the ipywidgets library to create the control interface for the program. Though it's the main function of the program it's main purpose is to call upon the other functions to do the calculations after the user has provided new settings for the simulation. Its secondary purpose is to run a calculation that can be turned on or off to calculate the break-even percentage of mass that needs to be removed through topological optimization in order for the TM and AM paths to be equivalent in energy usage. The method used for this calculation is a simplification/variation of the Newton Method.

A simulation will be created with the same input values as the ones inputted by the user, the only different value will be the *mass_reduction_perc* which will be substituted by *mass_reduc_solver*, a variable that will initialize at value 0.1 and its value will be iterated upon and changed, depending on whether or not the Energy demand by AM is bigger or smaller than TM's. If it is, the value of *mass_reduc_solver* will increase by the present step size. Additionally if the change then overshoots the desired equilibrium value then step size will be cut in half in order to slowly converge on desired value.

3.3.4 Interface

Before looking at scenarios and results it is important to understand how the inputs are made and how the output is presented.

Input

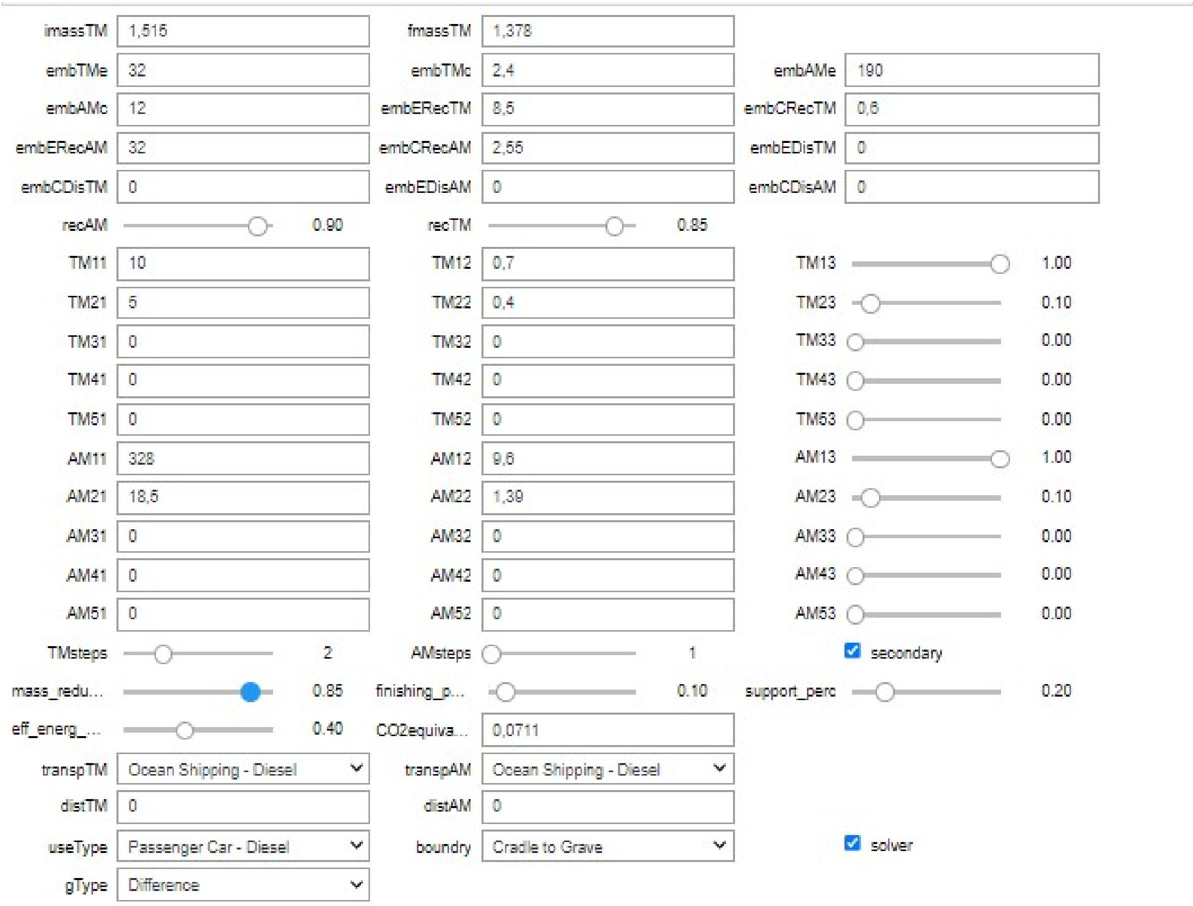


Figure 10: User input interface as it is presented on the program

On figure 11 you can see the user interface that is presented to the user when he runs the program. It consists of a series of text, sliders, dropdown menus and checkboxes as methods of inputting the necessary information for your program. How each variable is inputted and its description is explained on section 3.3.2.

The variables are separated into 6 groups that are displayed in sequence, in order to help the user find their desired variable. Each group has been squared off in order to help visualize the each one.

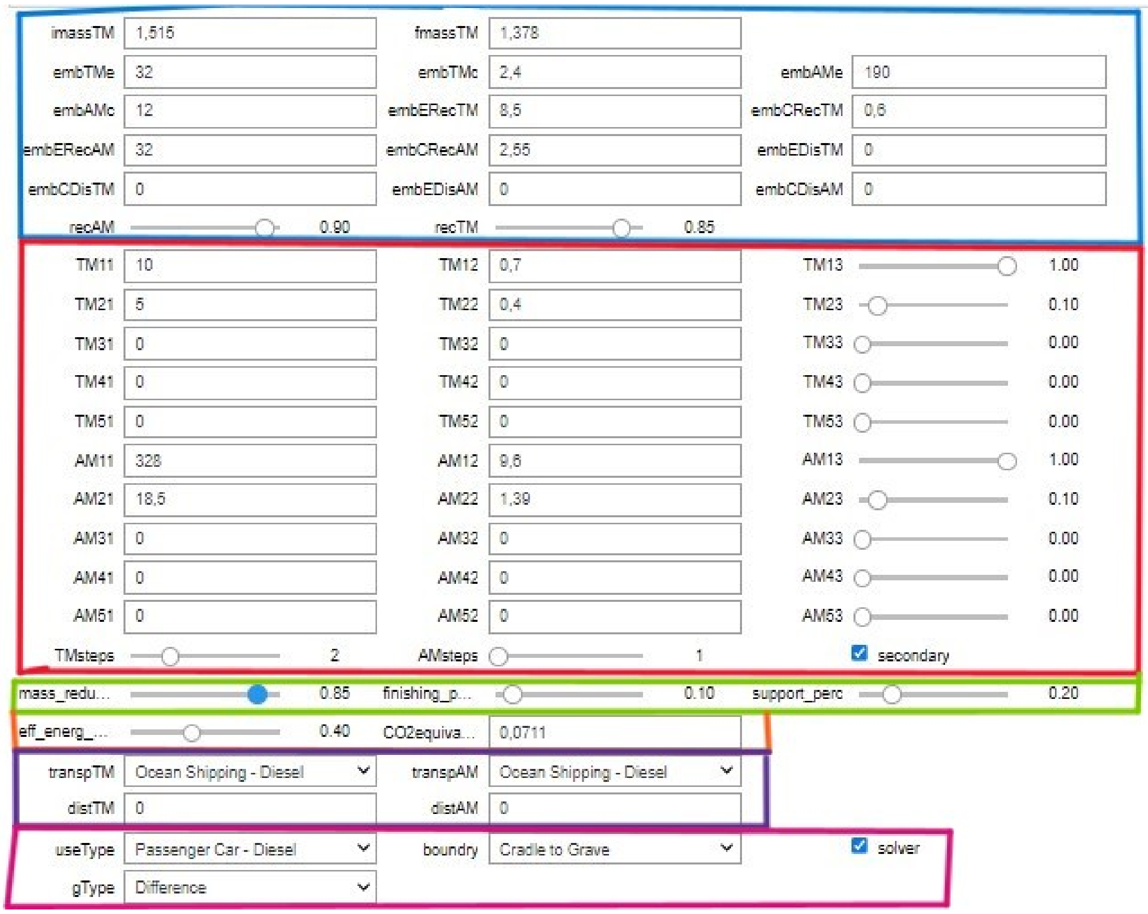


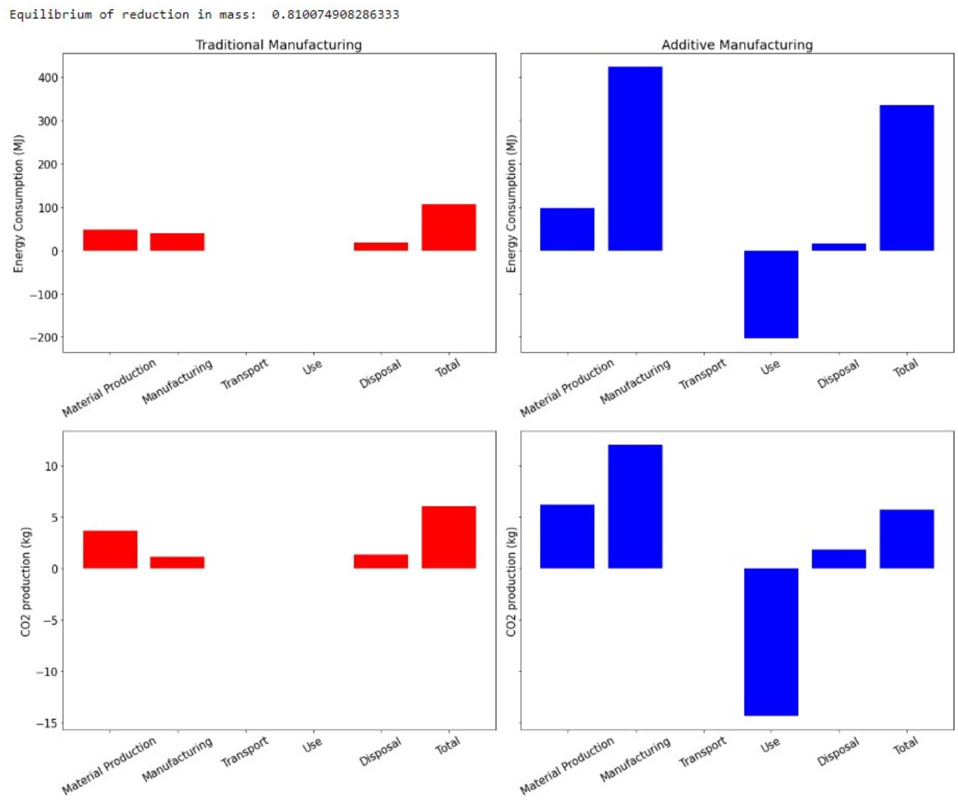
Figure 11: User input interface segmented into its six groups, from the top to the botom: Material, Manufacturing, Percentages, Power Grid, Transport and Control

The groups are the same as the ones used in section 3.3.2.

Output

In order to facilitate interpretation of the results presented in the output, this section will help explain where to find which information and how to read the graph in their two main possible formats.

Figure 12: Output: "All" and Solver result

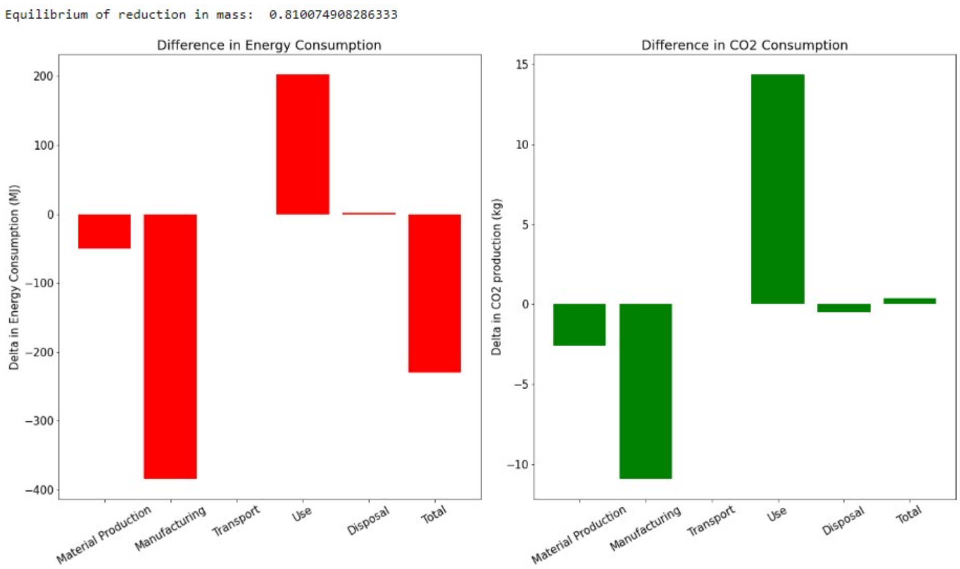


First we have the output presented for when *gType* is set as "All". In this mode, 4 graphs will be presented in a 2x2 arrangement, first column pertaining to Traditional Manufacturing results and second column pertaining to Additive Manufacturing results. The first row displays the graph in energy demand (MJ) and the second row displays the values in CO_2 emissions (kgCO_2).

When the boundary variable is set to Cradle to Grave, 6 bars will be present in each bar chart, they will be Material Production, Manufacturing, Transport, Use Disposal and Total respectively. Each will show the results of the simulated impact of their phase, except the total bar which will show the sum of all phases to facilitate comparisons between TM and AM.

One important thing to note is that in TM the Use phase will always have its results set to 0 since it is set as a baseline to compare to the impact "gains" offered by topological optimization in AM, which is why in AM the Use Phase will always be negative.

Figure 13: Output: "Difference" and Solver result



When *gType* is set to "Difference" a different visualization will be made in order to facilitate checking differences for each bar between TM and AM. Now two graphs will be displayed side by side, the one on the left will show the results in energy demand (MJ) and on the right they will be shown in emission of CO_2 (kgCO_2).

In this setting the values displayed are the difference between TM and AM. For each phase they are calculated in this manner:

$$D_{phase} = R_{TMphase} - R_{AMphase} \tag{3.6}$$

With D_{phase} meaning "Difference in this phase" and $R_{TMphase}$ and $R_{AMphase}$ representing "Results for this phase" for TM and AM. This means that if a bar is positive, as seen for example for the Use phase in figure 13 this means that the impact was higher for TM and when the bar is negative, the opposite is true, as can be seen in the manufacturing phase, meaning manufacturing impacts for AM were higher.

For both settings, "All" or "Difference", on the top left of the output the result of the solver that shows how much the mass reduction (variable *mass_reduction_perc* in the program) for AM has to be in order to have the same energy demands as TM in total for the boundary selected.

```
Energy:

TM
 {'Material Production': 48.48, 'Manufacturing': 39.76875, 'Transport': 0.0, 'Use': 0, 'Disposal': 18.217875, 'Total': 106.466
625}
AM
 {'Material Production': 98.1825, 'Manufacturing': 423.735, 'Transport': 0.0, 'Use': -202.56599999999997, 'Disposal': 16.536,
'Total': 335.88750000000005}
Carbon:

TM
 {'Material Production': 3.6359999999999997, 'Manufacturing': 1.13102325, 'Transport': 0.0, 'Use': 0, 'Disposal': 1.3180500000
000002, 'Total': 6.08507325}
AM
 {'Material Production': 6.2010000000000005, 'Manufacturing': 12.0510234, 'Transport': 0.0, 'Use': -14.372539999999999, 'Dispo
sal': 1.80604125, 'Total': 5.68552465}
```

Save input

Figure 14: Output: Values for each phase and Save Input button

Finally the second part of the output has two sections, the first are the numerical results per phase per manufacturing path displayed in case the user wants to note them down or save them. Second is the "Save input" button that instructs the program to save inputs in a .txt file in the same folder as the program so that the user can easily reproduce result without having to go back and change all the settings manually every time.

3.4 Simulation Results

In this section the simulation output will be presented for three scenarios. The first one will be used as validation of the results obtained through the program, as it will reproduce values found in another paper, and the other two will serve as showcases of simulations one can make with the program. There are infinite possibilities that can be simulated by the user and therefore it is encouraged that the reader experiments himself to see the capabilities of the software.

3.4.1 Validation

Here the program will be validated by comparing results achieved in simulation with results from existing article (PRIARONE; CATALANO; SETTINERI, 2023)

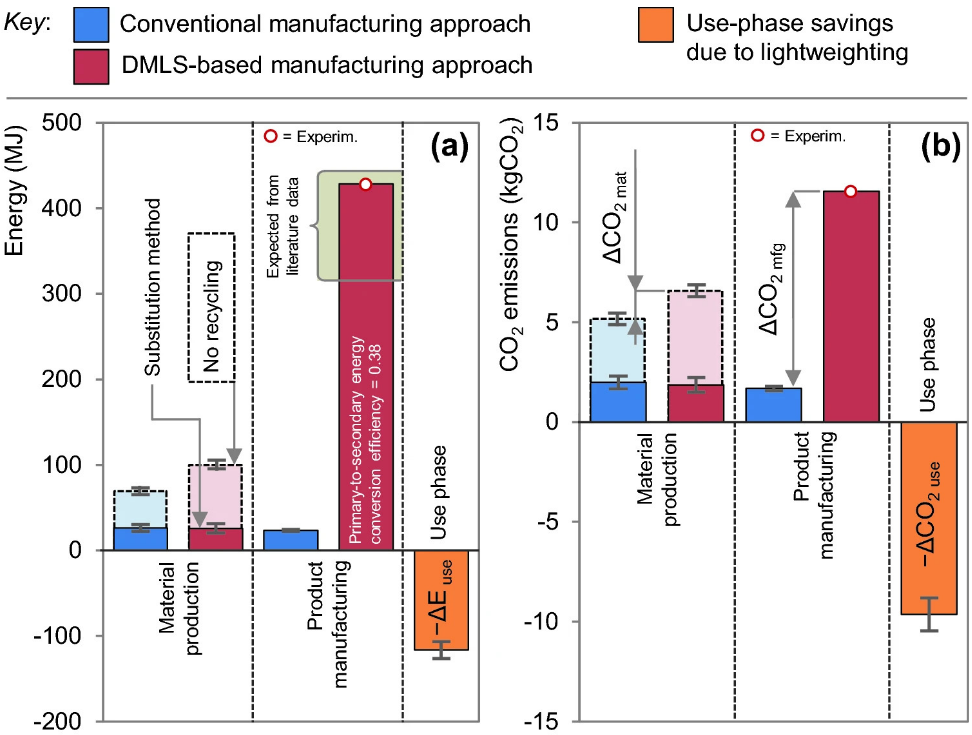


Figure 15: Results obtained in (PRIARONE; CATALANO; SETTINERI, 2023)

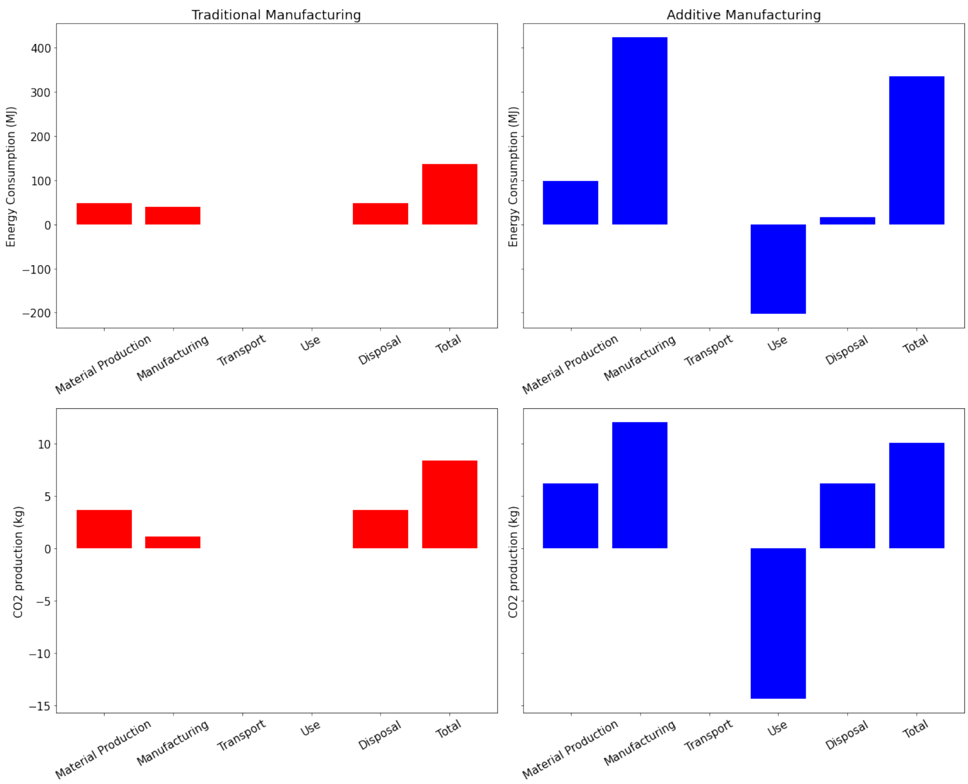


Figure 16: Results obtained in simulation

As can be seen on figures 15 and 16 the simulation does reproduce results achieved in other papers when utilizing the same input values. Some difference can be observed in the Use phase, that is due to an approximation done in the Program, as the closest option available to what was used in the article as the Use phase scenario was "Passenger Car - Diesel". In order to compare Material Production results, recyclability for both materials were set to 0 and as can be seen on both figures value do line up.

It is interesting to note how even though a 69% reduction in mass was achieved, it was still not enough to overcome the expressive energy expenditure necessary to create the part in the manufacturing phase. The percentage of reduction in mass would need to go up to 81% to reach break-even status. However CO_2 emissions were lower in AM than in TM.

The inputs used for the simulation can be seen on table 2.

Variable	**Input**
imassTM	1.515
fmassTM	1.378
embTMe	32
embTMc	2.4
embAMe	190
embAMc	12
embERecTM	8.5
embCRecTM	0.6
embERecAM	32
embCRecAM	2.55
embEDisTM	0
embCDisTM	0
embEDisAM	0
embCDisAM	0
recAM	0
recTM	0
TM11	10
TM12	0.7
TM13	1
TM21	5
TM22	0.4
TM23	0.1

Variable	Input
TM31	0
TM32	0
TM33	0
TM41	0
TM42	0
TM43	0
TM51	0
TM52	0
TM53	0
AM11	328
AM12	9.6
AM13	1
AM21	18.5
AM22	1.39
AM23	0.1
AM31	0
AM32	0
AM33	0
AM41	0
AM42	0
AM43	0
AM51	0
AM52	0
AM53	0
TMsteps	2
AMsteps	1
secondary	TRUE
mass_reduction_perc	0.7
finishing_perc	0
support_perc	0.2
eff_energ_conv	0.4
CO2equivalent	0.0711
distTM	0
distAM	0
useType	Passenger Car - Diesel

Variable	Input
boundary	Cradle to Grave

Table 2: Table of inputs used for validation scenario

3.4.2 Scenarios

Scenario 1 (Automotive Industry) - Variation of Validation Scenario

In this scenario some inputs have been altered from the validation scenario in order to simulate how the scenario might change as technology advances. Here are the changes that were made:

- Transport has now been considered to represent AM logistic advantages;
- Specific energy consumption for AM has been reduced by around 20% as a way to represent improvements in the technology allowing for better performance of the manufacturing phase;
- Embodied impact of material for AM increased by 10% to represent switch to new material with better properties allowing for mass reduction to grow to 79%;

The results obtained can be seen on figures 17 and 18.

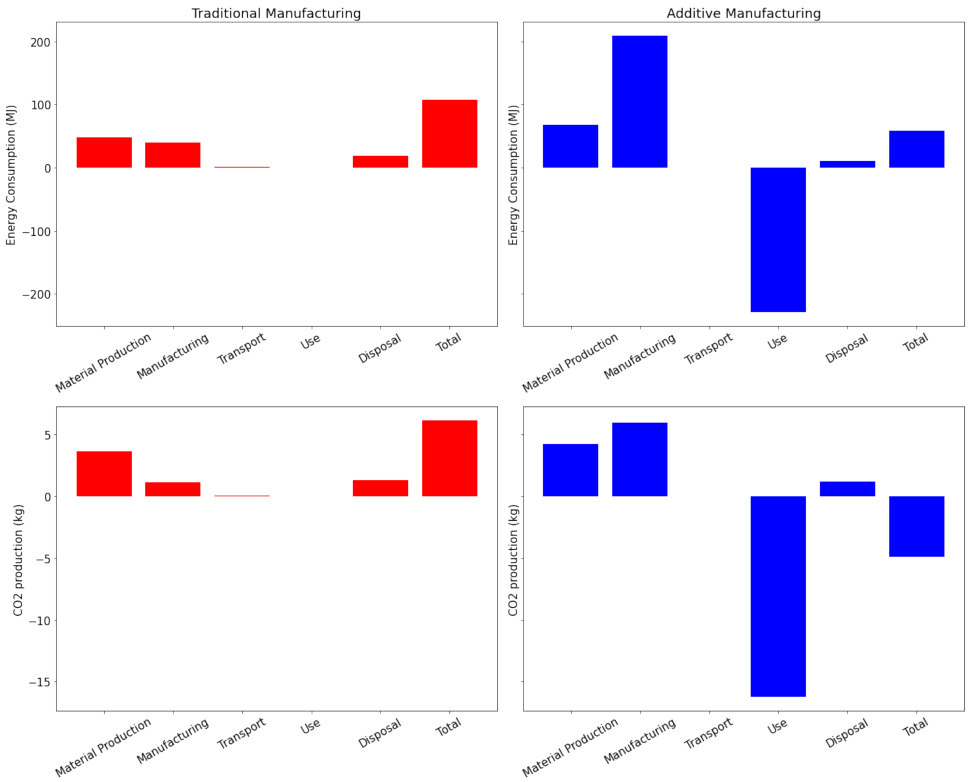


Figure 17: Graphs of Energy demand and CO2 emission for both Additive and Traditional Manufacturing under the input of Scenario 1

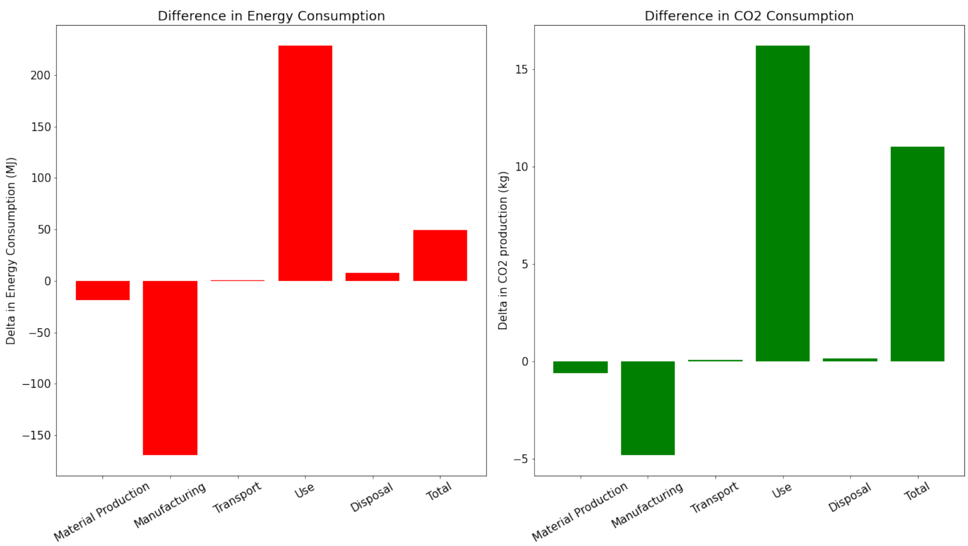


Figure 18: Graphs Delta in Energy demand and CO_2 emission between Additive and Traditional Manufacturing under the input of Scenario 1

The changes were enough to revert the previous energy advantage present for the TM path. Transport values were positive for AM in comparison to TM, however they were negligible in comparison to impacts of manufacturing and Use phase gains. As evidenced by the figure 18, manufacturing still remains as a much more impacting phase in AM when compared to TM, but the changes in material and further reduction of mass were able to outweigh those costs in the long run.

Inputs used can be seen on table 3.

Variable	**Input**
imassTM	1.515
fmassTM	1.378
embTMe	32
embTMc	2.4
embAMe	209
embAMc	13.2
embERecTM	8.5
embCRecTM	0.6
embERecAM	32
embCRecAM	2.55
embEDisTM	0
embCDisTM	0
embEDisAM	0
embCDisAM	0
recAM	0.9
recTM	0.85
TM11	10
TM12	0.7
TM13	1
TM21	5
TM22	0.4
TM23	0.1
TM31	0
TM32	0
TM33	0
TM41	0
TM42	0
TM43	0
TM51	0
TM52	0
TM53	0

Variable	Input
AM11	260
AM12	7.68
AM13	1
AM21	18.5
AM22	1.39
AM23	0.1
AM31	0
AM32	0
AM33	0
AM41	0
AM42	0
AM43	0
AM51	0
AM52	0
AM53	0
TMsteps	2
AMsteps	1
secondary	TRUE
mass_reduction_perc	0.79
finishing_perc	0
support_perc	0.1
eff_energ_conv	0.4
CO2equivalent	0.0711
transpTM	14 metric ton truck - Diesel
transpAM	14 metric ton truck - Diesel
distTM	400
distAM	10
useType	Passenger Car - Diesel
boundary	Cradle to Grave
solver	TRUE

Table 3: Table of inputs used for scenario 1

Scenario 2 (Aerospace Industry)

In this scenario the production of a part made out of Ti-6Al-4V will be considered. Here are some of the main considerations:

- In this scenario embodied energy of TM and AM are not so different, though the increased value for AM is justified due to atomization.
- Manufacturing phase will be composed of two steps for both AM and TM, the first step for TM will be a machining process that removes most of the mass needed and then undergoes a finishing step to remove the remaining mass. As for the AM, first step will consist of an Electron Beam Melting step that will create the shape of the part and then the part will undergo a finishing step as well to remove possible errors due to machine allowances.
- initial mass of the workpiece in TM will be 5.13kg, and final part mass will be 2.04kg. AM final part will be 0.45 kg constituting a mass reduction of around 80%, an additional 20% of mass will be placed due to support structures and extra 10% due to machine allowance.
- Selected use case type is Long-Distance Aircraft.

The results obtained can be seen on figures 19 and 20.

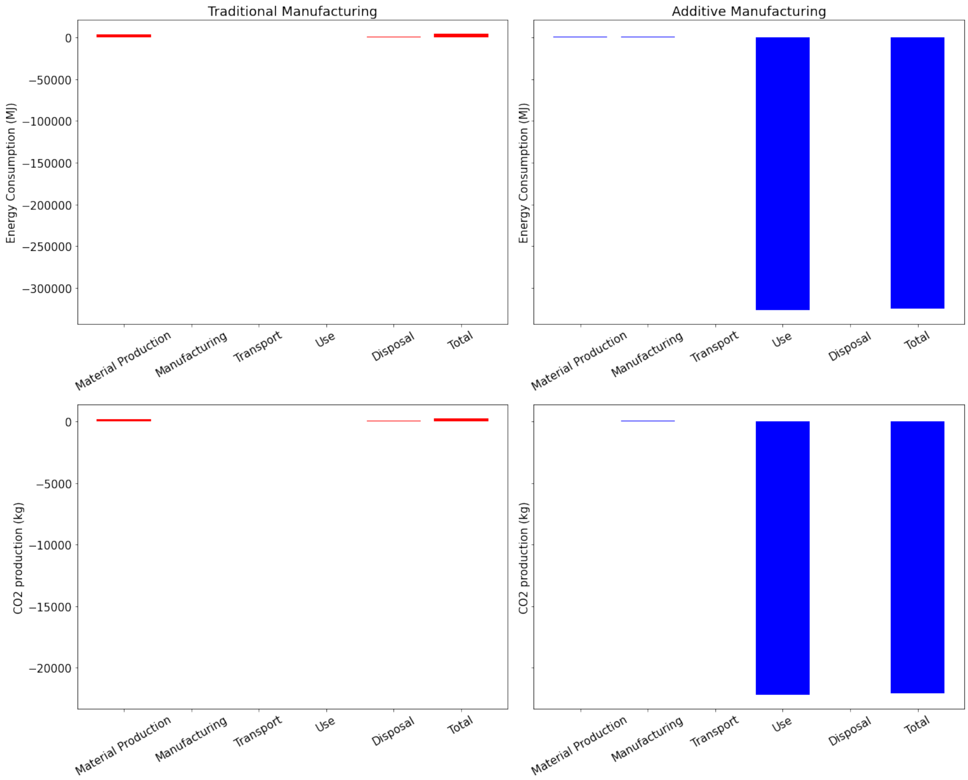


Figure 19: Graphs of Energy demand and CO2 emission for both Additive and Traditional Manufacturing under the input of Scenario 2.

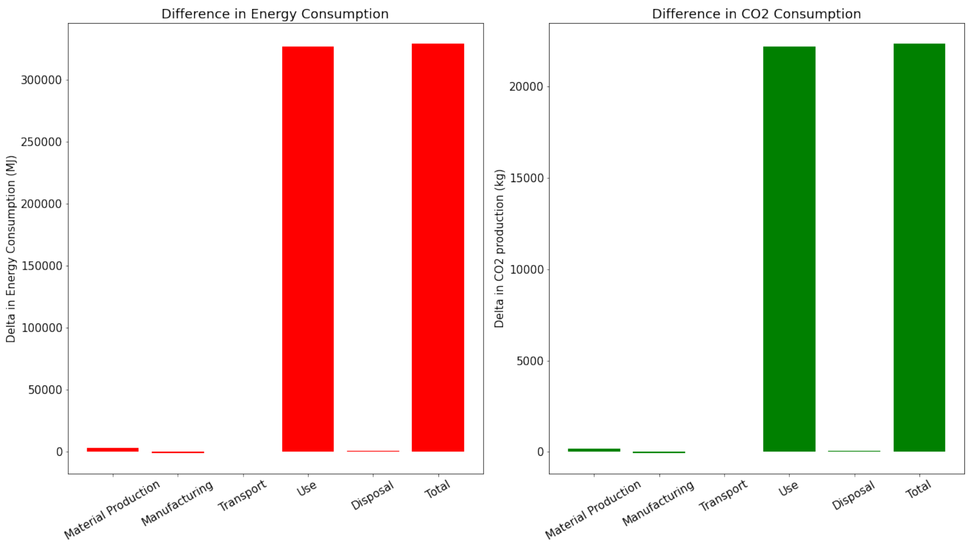


Figure 20: Graphs Delta in Energy demand and CO_2 emission between Additive and Traditional Manufacturing under the input of Scenario 2.

It might be confusing at first what the output is presenting and this is due to the impressive gains obtained by reducing mass in the aerospace industry. The value of the Use phase bar for AM is so negative (interpretation of graphs explained in section 3.3.4 under "Output") that it is almost impossible to see the impact of the other phases.

This simulation confirms the reason why AM has been adopted in the Aerospace industry, the gains of reducing mass outweigh every other impact to a point where other inputs/values are much less significant.

Inputs used can be seen on table 4.

Variable	Input
imassTM	5.13
fmassTM	2.04
embTMe	653
embTMc	39.4
embAMe	683
embAMc	40
embERecTM	8.5
embCRecTM	0.6
embERecAM	32
embCRecAM	2.55
embEDisTM	0
embCDisTM	0
embEDisAM	0
embCDisAM	0
recAM	0.8
recTM	0.75
TM11	2.28
TM12	0.17
TM13	0.52
TM21	18.5
TM22	1.39
TM23	0.1
TM31	0
TM32	0
TM33	0
TM41	0
TM42	0
TM43	0
TM51	0
TM52	0

Variable	**Input**
TM53	0
AM11	683
AM12	40
AM13	1
AM21	18.5
AM22	1.39
AM23	0.1
AM31	0
AM32	0
AM33	0
AM41	0
AM42	0
AM43	0
AM51	0
AM52	0
AM53	0
TMsteps	2
AMsteps	2
secondary	TRUE
mass_reduction_perc	0.8
finishing_perc	0.1
support_perc	0.2
eff_energ_conv	0.34
CO2equivalent	0.16
transpTM	14 metric ton truck - Diesel
transpAM	14 metric ton truck - Diesel
distTM	400
distAM	10
useType	Long-Distance Aircraft
boundary	Cradle to Grave
solver	TRUE

Table 4: Table of inputs used for scenario 2

4 CONCLUSIONS

In conclusion, Additive Manufacturing stands to be a disruptive technology that will fundamentally change the way our society produces nowadays. When referring back to the table 1 it seems that the scenario is not positive for the development of AM, as many disadvantages are stacked up on the Economic area, however, by researching the literature that was discussed throughout chapter 2, the biggest obstacles and barriers to the growth and adoption of this new manufacturing method are not long-term, they are not part of the nature of AM like the advantages. That is to say, the overcoming of these challenges are a matter of time and investment, since they are related specially with the fact that the technology is something that is new and starting to be developed and fully implemented. Whereas the advantages are characteristics that are intrinsic to the functionality of the AM processes: flexibility, greater design freedom, customization, reduction of waste and so forth. These are all things that are unattainable through TM, whereas the barriers for AM will be overcome after initial challenges are resolved as new research and investments are made.

The discussion is not whether or not Additive Manufacturing will overcome these challenges, but how quickly. Specially considering the relevance of this technology as it pertains to sustainable development and manufacturing. As described on chapter 1, *climate change is the defining issue of our time and we are at a defining moment*, of course, it is not as if Additive Manufacturing is the only development necessary to revert the situation, but would definitely be a part of the complete solution. The discussion is what efforts can be made to accelerate this adoption and transformation, so as to significantly reduce our environmental impact by avoiding unnecessary waste of resources and emissions of gases that harm the sustainability of the planet in the future. The purpose of the program is to facilitate the visualization and simulation of environmental gains allowed by AM, it has been validated on models where the dataset is available. It is open for use by other researchers, companies or anyone who's interested as it aims to function as a Decision Making tool that helps drive more investments into the technology, in order to hopefully allow the acceleration of adoption of the Additive Manufacturing, guaranteeing more sustainable manufacturing practices before it is too late.

REFERENCES

ABBEY, T. *What is topology optimization and why is it useful?* PTC, 2022. Disponível em: ⟨https://www.ptc.com/en/blogs/cad/what-is-topology-optimization#:~:text=Topology%20optimization%20takes%20a%203D,would%20normally%20use%20in%20design.⟩

ABDULHAMEED, O. et al. Additive manufacturing: Challenges, trends, and applications. *Advances in Mechanical Engineering*, SAGE Publications Sage UK: London, England, v. 11, n. 2, p. 1687814018822880, 2019.

ASHBY, M. F. *Materials and the environment: eco-informed material choice*. [S.l.]: Elsevier, 2012.

ATTARAN, M. The rise of 3-d printing: The advantages of additive manufacturing over traditional manufacturing. *Business Horizons*, Elsevier BV, v. 60, n. 5, p. 677–688, sep 2017.

BAUMERS, M. et al. The cost of additive manufacturing: machine productivity, economies of scale and technology-push. *Technological Forecasting and Social Change*, Elsevier BV, v. 102, p. 193–201, jan 2016.

BAUMERS, M. et al. How can material jetting systems be upgraded for more efficient multi-material additive manufacturing. In: UNIVERSITY OF TEXAS AT AUSTIN. *2014 International Solid Freeform Fabrication Symposium*. [S.l.], 2014.

BERMAN, B. 3-d printing: The new industrial revolution. *Business Horizons*, Elsevier BV, v. 55, n. 2, p. 155–162, mar 2012.

BöCKIN, D.; TILLMAN, A.-M. Environmental assessment of additive manufacturing in the automotive industry. *Journal of Cleaner Production*, Elsevier BV, v. 226, p. 977–987, jul 2019.

D'AVENI, R. A. 3-d printing will change the world. *Harvard business review*, Harvard Business School Publishing, v. 91, n. 3, p. 34–35, 2013.

DESPEISSE, M.; FORD, S. The role of additive manufacturing in improving resource efficiency and sustainability. In: UMEDA, S. et al. (Ed.). *Advances in Production Management Systems: Innovative Production Management Towards Sustainable Growth*. Cham: Springer International Publishing, 2015. p. 129–136. ISBN 978-3-319-22759-7.

DESPEISSE, M.; MINSHALL, T. Skills and education for additive manufacturing: a review of emerging issues. In: SPRINGER. *Advances in Production Management Systems. The Path to Intelligent, Collaborative and Sustainable Manufacturing: IFIP WG 5.7 International Conference, APMS 2017, Hamburg, Germany, September 3-7, 2017, Proceedings, Part I*. [S.l.], 2017. p. 289–297.

DWIVEDI, G.; SRIVASTAVA, S. K.; SRIVASTAVA, R. K. Analysis of barriers to implement additive manufacturing technology in the indian automotive sector. *International Journal of Physical Distribution & Logistics Management*, Emerald, v. 47, n. 10, p. 972–991, nov 2017.

EOS-GMBH. Accessed on February 07, 2023. Disponível em: ⟨https://www.eos.info/en/innovations/all-3d-printing-applications/aerospace/aerospace-case-studies/airbus-a350-xwb-3d-printed-cable-mount⟩.

FORD, S.; DESPEISSE, M. Additive manufacturing and sustainability: an exploratory study of the advantages and challenges. *Journal of Cleaner Production*, v. 137, p. 1573–1587, 2016. ISSN 0959-6526. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0959652616304395⟩.

Fratila, Domnita; Rotaru, Horatiu. Additive manufacturing - a sustainable manufacturing route. *MATEC Web Conf.*, v. 94, p. 03004, 2017. Disponível em: ⟨https://doi.org/10.1051/matecconf/20179403004⟩.

GARCIA, F. L. et al. Environmental performance of additive manufacturing process – an overview. *Rapid Prototyping Journal*, Emerald, v. 24, n. 7, p. 1166–1177, oct 2018.

GARCIA, F. L. et al. Comparative lca of conventional manufacturing vs. additive manufacturing: the case of injection moulding for recycled polymers. *International Journal of Sustainable Engineering*, Taylor & Francis, v. 14, n. 6, p. 1604–1622, 2021. Disponível em: ⟨https://doi.org/10.1080/19397038.2021.1990435⟩.

GIBSON, I. The changing face of additive manufacturing. *Journal of Manufacturing Technology Management*, Emerald, v. 28, n. 1, p. 10–17, feb 2017.

GIOVANNINI, A. et al. Ontology-based system for supporting manufacturing sustainability. *Annual Reviews in Control*, v. 36, n. 2, p. 309–317, 2012. ISSN 1367-5788. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S1367578812000478⟩.

GRASSI, G.; SPAGNOLO, S. L.; PAOLETTI, I. Fabrication and durability testing of a 3d printed façade for desert climates. *Additive Manufacturing*, Elsevier BV, v. 28, p. 439–444, aug 2019.

GUO, N.; LEU, M. C. Additive manufacturing: technology, applications and research needs. *Frontiers of Mechanical Engineering*, Springer Science and Business Media LLC, v. 8, n. 3, p. 215–243, may 2013.

GUTIERREZ-OSORIO, A. H. et al. Energy consumption analysis for additive manufacturing processes. *The International Journal of Advanced Manufacturing Technology*, Springer Science and Business Media LLC, v. 105, n. 1-4, p. 1735–1743, oct 2019.

GUTOWSKI, T. et al. Note on the rate and energy efficiency limits for additive manufacturing. *Journal of Industrial Ecology*, Wiley, v. 21, n. S1, p. S69–S79, oct 2017.

INGARAO, G. et al. Environmental modelling of aluminium based components manufacturing routes: Additive manufacturing versus machining versus forming. *Journal of Cleaner Production*, Elsevier BV, v. 176, p. 261–275, mar 2018.

International Standards Organization. *ISO 14040:2006: Environmental management — Life cycle assessment — Principles and framework*. [s.n.], 2006. Disponível em: ⟨https://www.iso.org/standard/37456.html⟩.

JAMSHIDINIA, M. et al. Additive manufacturing: of steel alloys using laser powder-bed fusion: in order to expand the choice of materials available for use in additive manufacturing, parameters that consider welding metallurgy, laser powder interaction, and post processing must be developed. *Advanced Materials & Processes*, ASM International, v. 173, n. 1, p. 20–25, 2015.

JAVAID, M.; HALEEM, A. Additive manufacturing applications in medical cases: A literature based review. *Alexandria Journal of Medicine*, Informa UK Limited, v. 54, n. 4, p. 411–422, dec 2018.

JIANG, J.; XU, X.; STRINGER, J. Support structures for additive manufacturing: A review. *Journal of Manufacturing and Materials Processing*, v. 2, n. 4, 2018. ISSN 2504-4494. Disponível em: ⟨https://www.mdpi.com/2504-4494/2/4/64⟩.

KARA, S.; LI, W. Unit process energy consumption models for material removal processes. *CIRP Annals*, Elsevier BV, v. 60, n. 1, p. 37–40, 2011.

KELLENS, K. et al. Environmental dimensions of additive manufacturing: mapping application domains and their environmental implications. *Journal of Industrial Ecology*, Wiley Online Library, v. 21, n. S1, p. S49–S68, 2017.

KIRCHHERR, J.; REIKE, D.; HEKKERT, M. Conceptualizing the circular economy: An analysis of 114 definitions. *Resources, Conservation and Recycling*, v. 127, p. 221–232, 2017. ISSN 0921-3449. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0921344917302835⟩.

KORHONEN, J.; HONKASALO, A.; SEPPäLä, J. Circular economy: The concept and its limitations. *Ecological Economics*, v. 143, p. 37–46, 2018. ISSN 0921-8009. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0921800916300325⟩.

LIFSET, R. 3d printing and industrial ecology. *Journal of Industrial Ecology*, Wiley, v. 21, n. S1, p. S6–S8, oct 2017.

MABKHOT, M. M. et al. Mapping industry 4.0 enabling technologies into united nations sustainability development goals. *Sustainability*, v. 13, n. 5, 2021. ISSN 2071-1050. Disponível em: ⟨https://www.mdpi.com/2071-1050/13/5/2560⟩.

MAISANO, D. A. et al. A structured comparison of decentralized additive manufacturing centers based on quality and sustainability. *The International Journal of Advanced Manufacturing Technology*, v. 121, n. 1, p. 993–1014, 2022. ISSN 1433-3015. Disponível em: ⟨https://doi.org/10.1007/s00170-022-09288-9⟩.

MANI, M.; LYONS, K. W.; GUPTA, S. K. Sustainability characterization for additive manufacturing. *Journal of research of the National Institute of Standards and Technology*, United States, v. 119, p. 419–28, 2014.

MELLOR, S.; HAO, L.; ZHANG, D. Additive manufacturing: A framework for implementation. *International Journal of Production Economics*, Elsevier BV, v. 149, p. 194–201, mar 2014.

MOLDAVSKA, A.; WELO, T. The concept of sustainable manufacturing and its definitions: A content-analysis based literature review. *Journal of Cleaner Production*, v. 166, p. 744–755, 2017. ISSN 0959-6526. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0959652617317213⟩.

NAJMON, J. C.; RAEISI, S.; TOVAR, A. Review of additive manufacturing technologies and applications in the aerospace industry. In: *Additive Manufacturing for the Aerospace Industry*. [S.l.]: Elsevier, 2019. p. 7–31.

NIAKI, M. K.; NONINO, F. Impact of additive manufacturing on business competitiveness: a multiple case study. *Journal of Manufacturing Technology Management*, Emerald, v. 28, n. 1, p. 56–74, feb 2017.

OLUYISOLA, O. E.; SGARBOSSA, F.; STRANDHAGEN, J. O. Smart production planning and control: Concept, use-cases and sustainability implications. *Sustainability*, v. 12, n. 9, 2020. ISSN 2071-1050. Disponível em: ⟨https://www.mdpi.com/2071-1050/12/9/3791⟩.

PECH, M.; VRCHOTA, J. The product customization process in relation to industry 4.0 and digitalization. *Processes*, MDPI AG, v. 10, n. 3, p. 539, mar 2022.

PENG, T. et al. Sustainability of additive manufacturing: An overview on its energy demand and environmental impact. *Additive Manufacturing*, v. 21, p. 694–704, 2018. ISSN 2214-8604. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S2214860417302646⟩.

PETLOCK Orthosis. Disponível em: ⟨https://hefestomedtech.com/petlock-orthosis⟩.

PRIARONE, P. C.; CATALANO, A. R.; SETTINERI, L. Additive manufacturing for the automotive industry: on the life-cycle environmental implications of material substitution and lightweighting through re-design. *Progress in Additive Manufacturing*, Jan 2023. ISSN 2363-9520. Disponível em: ⟨https://doi.org/10.1007/s40964-023-00395-x⟩.

PRIARONE, P. C.; INGARAO, G. Towards criteria for sustainable process selection: On the modelling of pure subtractive versus additive/subtractive integrated manufacturing approaches. *Journal of Cleaner Production*, Elsevier BV, v. 144, p. 57–68, feb 2017.

PRIARONE, P. C. et al. Influence of material-related aspects of additive and subtractive ti-6al-4v manufacturing on energy demand and carbon dioxide emissions. *Journal of Industrial Ecology*, Wiley, v. 21, n. S1, p. S191–S202, dec 2016.

REBITZER, G. et al. Life cycle assessment: Part 1: Framework, goal and scope definition, inventory analysis, and applications. *Environment International*, v. 30, n. 5, p. 701–720, 2004. ISSN 0160-4120. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0160412003002459⟩.

REN, L. et al. Graded biological materials and additive manufacturing technologies for producing bioinspired graded materials: An overview. *Composites Part B: Engineering*, Elsevier BV, v. 242, p. 110086, aug 2022.

ROSEN, M. A.; KISHAWY, H. A. Sustainable manufacturing and design: Concepts, practices and needs. *Sustainability*, v. 4, n. 2, p. 154–174, 2012. ISSN 2071-1050. Disponível em: ⟨https://www.mdpi.com/2071-1050/4/2/154⟩.

SAADE, M. R. M.; YAHIA, A.; AMOR, B. How has LCA been applied to 3d printing? a systematic literature review and recommendations for future studies. *Journal of Cleaner Production*, Elsevier BV, v. 244, p. 118803, jan 2020.

SAMYKANO, M. et al. Mechanical property of FDM printed ABS: influence of printing parameters. *The International Journal of Advanced Manufacturing Technology*, Springer Science and Business Media LLC, v. 102, n. 9-12, p. 2779–2796, feb 2019.

SEABRA, M. et al. Selective laser melting (SLM) and topology optimization for lighter aerospace componentes. *Procedia Structural Integrity*, Elsevier BV, v. 1, p. 289–296, 2016.

SLAPER, T. F.; HALL, T. J. et al. The triple bottom line: What is it and how does it work. *Indiana business review*, v. 86, n. 1, p. 4–8, 2011.

SREENIVASAN, R.; GOEL, A.; BOURELL, D. Sustainability issues in laser-based additive manufacturing. *Physics Procedia*, v. 5, p. 81–90, 2010. ISSN 1875-3892. Laser Assisted Net Shape Engineering 6, Proceedings of the LANE 2010, Part 1. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S187538921000550X⟩.

TADDESE, G.; DURIEUX, S.; DUC, E. Sustainability performance indicators for additive manufacturing: a literature review based on product life cycle studies. *The International Journal of Advanced Manufacturing Technology*, v. 107, n. 7, p. 3109–3134, 2020. ISSN 1433-3015. Disponível em: ⟨https://doi.org/10.1007/s00170-020-05249-2⟩.

THOMPSON, M. K. et al. Design for additive manufacturing: Trends, opportunities, considerations, and constraints. *CIRP annals*, Elsevier, v. 65, n. 2, p. 737–760, 2016.

VACCHI, M. et al. Technological sustainability or sustainable technology? a multidimensional vision of sustainability in manufacturing. *Sustainability*, v. 13, n. 17, 2021. ISSN 2071-1050. Disponível em: ⟨https://www.mdpi.com/2071-1050/13/17/9942⟩.

VITALE, M.; COTTELEER, M.; HOLDOWSKY, J. An overview of additive manufacturing. *Defense AT&L, November-December*, 2016.

WHAT is Numpy? ⟨https://numpy.org/doc/stable/user/whatisnumpy.html⟩. Accessed: 2023-01-31.

ZAWADZKI, P.; ŻYWICKI, K. Smart product design and production control for effective mass customization in the industry 4.0 concept. *Management and production engineering review*, Production Engineering Committee of the Polish Academy of Sciences, Polish . . . , 2016.

ZHU, J. et al. A review of topology optimization for additive manufacturing: Status and challenges. *Chinese Journal of Aeronautics*, Elsevier BV, v. 34, n. 1, p. 91–110, jan 2021.

ANNEX A – CODE

Decision Making LCA Tool

Welcome to the program! This program will help you simulate LCA scenarios for both Traditional Manufacturing and Additive Manufacturing and enable you to compare them

```
#### Importing necessary libraries

# Libraries for calculations
import numpy as np

from datetime import datetime as date
import functools

# Libraries for displaying information
import matplotlib.pyplot as plt
import ipywidgets
from ipywidgets import interactive,interact, HBox, Layout,VBox
from IPython.display import display
```

Defining Useful functions

LCA Functions

Material Production Phase

```
def MaterialProduction(EmbodiedMat1, EmbodiedMat2, massTM, massAM):
    '''
    Description:
    ------------

    This function will calculate material production energy
requirements and
    CO2 emissions for both TM and AM simultaneously by multiplying the
array
    of embodied energy and carbon by the mass relevant for the given
path

    Variables:
    ----------

    EmbodiedMat1:
    - Type: Array
    - Desc: Array with the embodied energy (MJ/kg) and carbon
(kgCO2/kg) of the
    material used in TM,  - [Energy, Carbon].

    EmbodiedMat2:
    - Type: Array
    - Desc: Array with the embodied energy (MJ/kg) and carbon
```

```
(kgCO2/kg) of the
    material used in AM,  - [Energy, Carbon].

    massTM:
    - Type: Float
    - Desc: Stores the mass of the initial workpiece that will be used
in TM path.

    massAM:
    - Type: Float
    - Desc: Stores the mass of the mass that will be placed on the AM
path (has
    to consider support mass and allowance of the machine)

    Returns:
    --------

    EG_TM:
    - Type: Array
    - Desc: Stores the energy and carbon demand for the material
production for
    the workpiece in TM

    EG_AM:
    - Type: Array
    - Desc: Stores the energy and carbon demand for the material
production of
    the material that will be placed in AM


    '''
    # Simply mutiplying the array of embodied impacts by the initial
mass of both production paths.
    EG_TM = np.multiply(EmbodiedMat1,massTM)
    EG_AM = np.multiply(EmbodiedMat2,massAM)
    return EG_TM, EG_AM
```

Manufacturing Phase

```
def Manufacturing(processing_steps):
    '''
    Description:
    ------------

    This function will calculate the energy and carbon cost of
manufacturing in
    each of the manufacturing paths by multiplying the Energy demand
(MJ/kg) and
    carbon emission (kgCO2/kg) for each step by the mass that will be
removed/placed in each step
```

```
    Variables:
    ----------

    processing_steps:
    - Type: Array

    - Desc: Each of the rows in the array ius meant to represent one
step in the manufacturing
    process, for example, if you have a milling step then a finishing
step, the first row will
    relate to the costs and mass removed of the milling step and the
finishing step will be described
    on the second row - it`s done in this format -
                                    [[Specific Energy Demand, Specific
CO2 Emission, Mass],
                                     [Specific Energy Demand, Specific
CO2 Emission, Mass],
                                     [Specific Energy Demand, Specific
CO2 Emission, Mass],
                                     [Specific Energy Demand, Specific
CO2 Emission, Mass],
                                     [Specific Energy Demand, Specific
CO2 Emission, Mass]]


    Returns:
    --------

    result:
    - Type: Array in this format ["Total Energy demand", "Total kgCO2
emitted"]
    - Desc: Stores the result of the multiplication and sum of
specific impacts for the manufacturing steps


    '''


    # Multiplies the specific impacts (that are placed on the first
two columns) by the mass relevant for
    # that step (in the third column)
    result = np.multiply(processing_steps[:,:2],processing_steps[:,
[2]])

    # Sums the total for each column, achieving the total energy
demand and CO2 emission for the whole manufacturing phase
    result = np.sum(result, axis=0)

    return result
```

Transport Phase

```
# def Transport(distanceAM, distanceTM, methodAM, methodTM):
def Transport(mass, distance, method):
    '''
    Description:
    ------------
    Calculates the impacts of transportation of the manufactured 
produt until
    the use phase. Simbolizes the distance travelled until an assembly 
step at
    another place


    Variables:
    ----------
    mass:
    - Type: Float

    - Desc: Simply stores the mass of the product that will be 
transported

    distance:
    - Type: Float

    - Desc: Stores the distance that will be travelled between 
manufacturing and
    use phase

    method:
    - Type: String

    - Desc: Stores the method of transportation, which then will be 
used as a key for a
    dictionary in the fucntion to gather the transportation impacts 
per metric ton * km

    Returns:
    --------
    result:
    - Type: Array

    - Desc: Stores the result of the multiplication of mass, distance 
and specific transport impacts




    '''
    # DICTIONARY - MJ PER (METRIC TON*KM)
    energyDemand = {'Ocean Shipping - Diesel': 0.16,
```

```
                    'Coastal Shipping - Diesel': 0.27,
                    'Barge - Diesel': 0.36,
                    'Rail - Diesel': 0.25,
                    'Articulated HGV (up to 55 metric tons) - Diesel':
0.71,
                    '40 metric ton truck - Diesel': 0.82,
                    '32 metric ton truck - Diesel': 0.94,
                    '14 metric ton truck - Diesel': 1.5,
                    'Light goods vehicle - Diesel': 2.5,
                    'Family car - Diesel': 1.7,
                    'Family car - Gasoline': 2.6,
                    'Family car -LPG': 3.9,
                    'Family car - Hybrid Gasoline-electric': 1.55,
                    'Super sports car and SUV - Gasoline': 4.8,
                    'Long Haul Aircraft - Kerosene': 6.5,
                    'Short Haul Aircraft - Kerosene': 13,
                    'Helicopter (Eurocopter AS350) - Kerosene': 55}

    # DICTIONARY - KG OF CO2 PER (METRIC TON*KM)
    CO2Demand = {'Ocean Shipping - Diesel': 0.015,
                'Coastal Shipping - Diesel': 0.019,
                'Barge - Diesel': 0.028,
                'Rail - Diesel': 0.019,
                'Articulated HGV (up to 55 metric tons) - Diesel':
0.05,
                '40 metric ton truck - Diesel': 0.06,
                '32 metric ton truck - Diesel': 0.067,
                '14 metric ton truck - Diesel': 0.11,
                'Light goods vehicle - Diesel': 0.18,
                'Family car - Diesel': 0.12,
                'Family car - Gasoline': 0.165,
                'Family car -LPG': 0.18,
                'Family car - Hybrid Gasoline-electric': 0.10,
                'Super sports car and SUV - Gasoline': 0.31,
                'Long Haul Aircraft - Kerosene': 0.45,
                'Short Haul Aircraft - Kerosene': 0.76,
                'Helicopter (Eurocopter AS350) - Kerosene': 3.3}

    # Initializing array with transport impacts using method as a key
    transportDemands = [energyDemand[method],CO2Demand[method]]

    # Multiplying the impacts by the products of mass and distance
    # (important to note that since impacts are in metric tons we have

    # to divide mass by 1000 since they are in kg)
    result = np.multiply(transportDemands,(mass/1000)*distance)

    return result
```

Use Phase

```
def Use(useType, dif_mass):
    '''
    Description:
    ------------

    This function will calculate the "gain" (the decrease of
environmental cost
    due to bigger efficiecies in geometry in AM) in energy demand and
carbon emissions
    when the produced piece is in use.

    Variables:
    ----------

    useType:
    - Type: String
    - Desc: Will be used as a key to the dictionaries of energy
savings and carbon demand
    decrease per kg of decreased weight

    dif_mass:
    - Type: float
    - Desc: Stores the difference in mass between the mass of the
finalized product
    manufactured through TM and AM


    Returns:
    --------

    result:
    - Type: Array
    - Desc: Stores the multiplicaion of the energy and a carbon gains
in an array in
    this format- [Energy Savings, CO2 savings]


    '''

    # DICTIONARY - MJ PER KG OF REDUCED WEIGHT
    energySaving = {'Long-Distance Aircraft': 200*(10**3),'Short-
Distance Aircraft': 150*(10**3),
                    'Articulated Truck': 260,'Passenger Car - Diesel':
210, 'Passenger Car - Gasoline': 230}

    # DICTIONARY - KG OF CO2 PER KG OF REDUCED WEIGHT
    CO2Saving = {'Long-Distance Aircraft': 13.6*(10**3),'Short-
Distance Aircraft': 10.2*(10**3),
                    'Articulated Truck': 18.5,'Passenger Car -
```

```
Diesel': 14.9, 'Passenger Car - Gasoline': 15}


    # Creating the array that will be multiplied by the difference in
mass, by using the key on the dictionaries
    valuesUse = [energySaving[useType],CO2Saving[useType]]

    # Mutiplies the array by the difference in mass arriving at the
output
    result = np.multiply(valuesUse, dif_mass)

    return result
```

Disposal Phase

```
def Disposal(recyclability, emb_virgin, emb_waste, emb_rec, mass):
    '''
    Description:
    ------------

    Using the "Substitution Method" the environmental impact of the
disposal is calculated

    Variables:
    ----------

    recyclability:
    - Type: float
    - Desc: % of recyclability (combined recovery, recycling rate and
recycling mateiral yield)

    emb_virgin:
    - Type: Float
    - Desc: Stores the embodied impact from virgin material input per
unit of material

    emb_waste:
    - Type: Float
    - Desc: Stores the embodied impact for disposal of material per
unit of material

    emb_rec:
    - Type: Float
    - Desc: Stores the embodied impact from recycled material input
per unit of material

    mass:
    - Type: Float
    - Desc: Stores the mass that is being disposed of

    Returns:
```

```
    --------

    result:
    - Type: Array
    - Desc: Stores the result of the substitution method in this
format: [Energy demand, CO2 emission]




    '''


    # Applying the substitution method to calculate the embodied
impact of disposal
    E = np.multiply(emb_rec,recyclability)
+np.multiply((emb_virgin+emb_waste),1-recyclability)

    # Multiplying array by the mass that goes to disposal
    result = np.multiply(E,mass)

    return result
```

Control Functions

Calculation

LCA Function, simulates the LCA by calling each of the phase fucntions accordingly

```
def Calculation(imassTM,
                fmassTM,
                embTMe,
                embTMc,
                embAMe,
                embAMc,
                embERecTM,
                embCRecTM,
                embERecAM,
                embCRecAM,
                embEDisTM,
                embCDisTM,
                embEDisAM,
                embCDisAM,
                recAM,
                recTM,
                TM11 ,
                TM12 ,
                TM13 ,
                TM21 ,
```

```
                TM22 ,
                TM23,
                TM31,
                TM32,
                TM33,
                TM41,
                TM42,
                TM43,
                TM51,
                TM52,
                TM53,
                AM11,
                AM12,
                AM13,
                AM21,
                AM22,
                AM23,
                AM31,
                AM32,
                AM33,
                AM41,
                AM42,
                AM43,
                AM51,
                AM52,
                AM53,
                TMsteps,
                AMsteps,
                secondary,
                mass_reduction_perc,
                finishing_perc,
                support_perc,
                eff_energ_conv,
                CO2equivalent,
                transpTM ,
                transpAM ,
                distTM ,
                distAM ,
                useType,
                boundary):
    '''
    Description:
    ------------

    This is the core function of the program as it pertains to the
calculation,
    it controls and calls all the other functions into action. It has
been separated
    from the main function since it is called more than once, and in
order to be more
```

```
    efficient in code it was separated

    Variables:
    ----------

    imassTM:
    - Type: float
    - Desc: Stores the initial mass of the TM workpiece
    fmassTM:
    - Type: float
    - Desc: Stores the final mass after the workpiece has been
finished in TM
    embTMe:
    - Type: float
    - Desc: Stores the embodied energy impact for the material per
unit of material
    used in TM
    embTMc:
    - Type: float
    - Desc: Stores the embodied CO2 impact for the material per unit
of material
    used in TM
    embAMe:
    - Type: float
    - Desc: Stores the embodied energy impact for the material per
unit of material
    used in AM
    embAMc:
    - Type: float
    - Desc: Stores the embodied CO2 impact for the material per unit
of material
    used in AM
    embEDisTM:
    - Type: Float
    - Desc: Stores the embodied energy for disposal of material per
unit of material used in TM
    embCDisTM:
    - Type: Float
    - Desc: Stores the embodied carbon for disposal of material per
unit of material used in TM
    embEDisAM:
    - Type: Float
    - Desc: Stores the embodied energy for disposal of material per
unit of material used in AM
    embCDisAM:
    - Type: Float
    - Desc: Stores the embodied carbon for disposal of material per
unit of material used in AM
    embERecTM:
    - Type: Float
```

```
    - Desc: Stores the embodied energy impact from recycled material
input per unit of material used in TM
    embCRecTM:
    - Type: Float
    - Desc: Stores the embodied carbon from recycled material input
per unit of material used in TM
    embERecAM:
    - Type: Float
    - Desc: Stores the embodied energy impact from recycled material
input per unit of material used in AM
    embCRecAM:
    - Type: Float
    - Desc: Stores the embodied carbon from recycled material input
per unit of material used in AM
    TM11:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
first step
    of the TM path
    TM12:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
first step
    of TM path
    TM13:
    - Type: float
    - Desc: Stores the mass that will be removed in first step of the
TM path
    TM21:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
second step
    of the TM path
    TM22:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
second step
    of TM path
    TM23:
    - Type: float
    - Desc: Stores the mass that will be removed in second step of the
TM path
    TM31:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
third step
    of the TM path
    TM32:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
```

```
third step
    of TM path
    TM33:
    - Type: float
    - Desc: Stores the mass that will be removed in third step of the
TM path
    TM41:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
fourth step
    of the TM path
    TM42:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
fourth step
    of TM path
    TM43:
    - Type: float
    - Desc: Stores the mass that will be removed in fourth step of the
TM path
    TM51:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
fifth step
    of the TM path
    TM52:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
fifth step
    of TM path
    TM53:
    - Type: float
    - Desc: Stores the mass that will be removed in fifth step of the
TM path
    TMsteps:
    - Type: float
    - Desc: Stores the ammount of steps that shpuld be considerd in
this
    simulation for the TM path, as in if the number 3 is selected 3
    steps will be considered, if 2 is selected only the first two
steps will
    be considered. Can go up to 5
    AM11:
    - Type: float
    - Desc: Stores the energy usage per unit of material
removed/placed in first step
    of the AM path
    AM12:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
```

```
first step
    of AM path
    AM13:
    - Type: float
    - Desc: Stores the mass that will be removed/placed in first step
of the AM path
    AM21:
    - Type: float
    - Desc: Stores the energy usage per unit of material
removed/placed in second step
    of the AM path
    AM22:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
second step
    of AM path
    AM23:
    - Type: float
    - Desc: Stores the mass that will be removed/placed in second step
of the AM path
    AM31:
    - Type: float
    - Desc: Stores the energy usage per unit of material
removed/placed in third step
    of the AM path
    AM32:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
third step
    of AM path
    AM33:
    - Type: float
    - Desc: Stores the mass that will be removed/placed in third step
of the AM path
    AM41:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
fourth step
    of the AM path
    AM42:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
fourth step
    of AM path
    AM43:
    - Type: float
    - Desc: Stores the mass that will be removed in fourth step of the
AM path
    TM51:
    - Type: float
```

```
    - Desc: Stores the energy usage per unit of material removed in
fifth step
    of the AM path
    AM52:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
fifth step
    of AM path
    AM53:
    - Type: float
    - Desc: Stores the mass that will be removed in fifth step of the
AM path
    AMsteps:
    - Type: float
    - Desc: Stores the ammount of steps that shpuld be considerd in
this
    simulation for the AM path, as in if the number 3 is selected all
3
    steps will be considered, if 2 is selected only the first two
steps will
    be considered
    mass_reduction_perc:
    - Type: float
    - Desc: Percentage of the final mass of the product produced
through TM,
    that will be reduced due to geometry optimization in AM path
    finishing_perc:
    - Type: float
    - Desc: Percentage of additional mass that is placed in the AM
process due
    to machine allowance that will later need to be removed
    support_perc:
    - Type: float
    - Desc: Percentage of additional mass that is placed in the AM
process due
    to the need to create supports during the build of a particular
piece
    eff_energ_conv:
    - Type: float
    - Desc: Stores the efficiency of the Power Grid considered in the
simulation
    CO2equivalent:
    - Type: float
    - Desc: Stores the equivalent CO2 emissions due to the energy
required to
    produce the piece in the considered Power Grid in the simulation


    recTM:
    - Type: float
```

```
    - Desc: % of recyclability (combined recovery, recycling rate and
recycling mateiral yield)


    recAM:
    - Type: float
    - Desc: % of recyclability (combined recovery, recycling rate and
recycling mateiral yield)

    secondary:
    - Type: boolean
    - Desc: Serves as a flag of whether or no the input for
manufacturing steps is in primary or secondary use

    useType:
    - Type: string
    - Desc: Stores the input of where the user wants to consider
thepiece to be used in;

    transpTM:
    - Type: string
    - Desc: Stores the name of the transportation mode that will be
used for TM, serves as a key for
    the dictionary of specific impacts

    transpAM:
    - Type: string
    - Desc: Stores the name of the transportation mode that will be
used for AM, serves as a key for
    the dictionary of specific impacts

    distTM:
    - Type: float
    - Desc:  Stores the distance that will be travelled between
manufacturing and
    use phase in TM path

    distAM:
    - Type: float
    - Desc: Stores the distance that will be travelled between
manufacturing and
    use phase in AM path

    boundary:
    - Type: String
    - Desc: Stores boundary that will be considered for this
simulation, depending
    on its value graphs will display either a Cradle to grave, or
Cradle to Gate or Gate to Gate scenario
```

```
    Returns:
    --------

    EnergyBarsTM:
    - Type: dictionary
    - Desc: Stores the energy impact result of each of the life cycle
phases of the TM path

    CO2BarsTM:
    - Type: dictionary
    - Desc: Stores the CO2 impact result of each of the life cycle
phases of the TM path

    EnergyBarsAM:
    - Type: dictionary
    - Desc: Stores the energy impact result of each of the life cycle
phases of the AM path

    CO2BarsAM:
    - Type: dictionary
    - Desc: Stores the CO2 impact result of each of the life cycle
phases of the AM path




    '''

    ############## MATERIAL PRODUCTION

    massAM = fmassTM*(1-mass_reduction_perc)/(1-
(support_perc+finishing_perc))

    TMmat_prod, AMmat_prod = MaterialProduction([embTMe,embTMc] ,
[embAMe,embAMc] , imassTM, massAM)

    ############# MANUFACTURING

    removed_massTM = imassTM-fmassTM
    if secondary == False:
        TM_processing = np.array([[TM11,TM12,TM13*imassTM],
[TM21,TM22,TM23*imassTM],[TM31,TM32,TM33*imassTM],
[TM41,TM42,TM43*imassTM],[TM51,TM52,TM53*imassTM]])
        uTM_processing = TM_processing
        uTM_processing[TMsteps:] = 0

        AM_processing = np.array([[AM11,AM12,AM13*massAM],
[AM21,AM22,AM23*massAM],[AM31,AM32,AM33*massAM],
[AM41,AM42,AM43*massAM],[AM51,AM52,AM53*massAM]])
```

```
        uAM_processing = AM_processing
        uAM_processing[AMsteps:] = 0

    else:
        EnergyProcessTM =
np.divide([TM11,TM21,TM31,TM41,TM51],eff_energ_conv)
        CO2ProcessTM =
np.multiply([TM11,TM21,TM31,TM41,TM51],CO2equivalent)
        EnergyProcessAM =
np.divide([AM11,AM21,AM31,AM41,AM51],eff_energ_conv)
        CO2ProcessAM =
np.multiply([AM11,AM21,AM31,AM41,AM51],CO2equivalent)

        TM_processing =
np.array([[EnergyProcessTM[0],CO2ProcessTM[0],TM13*imassTM],

[EnergyProcessTM[1],CO2ProcessTM[1],TM23*imassTM],

[EnergyProcessTM[2],CO2ProcessTM[2],TM33*imassTM],

[EnergyProcessTM[3],CO2ProcessTM[3],TM43*imassTM],

[EnergyProcessTM[4],CO2ProcessTM[4],TM53*imassTM]])
        uTM_processing = TM_processing
        uTM_processing[TMsteps:] = 0

        AM_processing =
np.array([[EnergyProcessAM[0],CO2ProcessAM[0],AM13*massAM],

[EnergyProcessAM[1],CO2ProcessAM[1],AM23*massAM],

[EnergyProcessAM[2],CO2ProcessAM[2],AM33*massAM],

[EnergyProcessAM[3],CO2ProcessAM[3],AM43*massAM],

[EnergyProcessAM[4],CO2ProcessAM[4],AM53*massAM]])
        uAM_processing = AM_processing
        uAM_processing[AMsteps:] = 0



    TMmanuf = Manufacturing(uTM_processing)
    AMmanuf = Manufacturing(uAM_processing)

    ############# TRANSPORT

    TMtransport = Transport(fmassTM, distTM, transpTM)
    AMtransport = Transport(fmassTM*(1-mass_reduction_perc), distAM,
transpAM)
```

```
    ############# USE

    reducedWeight = fmassTM*(-mass_reduction_perc)
    AMuse = Use(useType, reducedWeight)

    ############# DISPOSAL


    TMdisposal = 
Disposal(recTM,np.array([embTMe,embTMc]),np.array([embEDisTM,embCDisTM
]),np.array([embERecTM,embCRecTM]),imassTM)
    AMdisposal = 
Disposal(recAM,np.array([embTMe,embAMc]),np.array([embEDisAM,embCDisAM
]),np.array([embERecAM,embCRecAM]),massAM)





    if boundary == 'Cradle to Grave':

        EnergyBarsTM  = {'Material Production':TMmat_prod[0], 
'Manufacturing':TMmanuf[0], 'Transport':TMtransport[0], 'Use':0, 
'Disposal':TMdisposal[0]}
        CO2BarsTM = {'Material Production':TMmat_prod[1], 
'Manufacturing':TMmanuf[1], 'Transport':TMtransport[1], 'Use':0, 
'Disposal':TMdisposal[1]}

        EnergyBarsTM["Total"] = sum(EnergyBarsTM.values())
        CO2BarsTM["Total"] = sum(CO2BarsTM.values())

        EnergyBarsAM  = {'Material Production':AMmat_prod[0], 
'Manufacturing':AMmanuf[0], 'Transport':AMtransport[0], 
'Use':AMuse[0], 'Disposal':AMdisposal[0]}
        CO2BarsAM = {'Material Production':AMmat_prod[1], 
'Manufacturing':AMmanuf[1], 'Transport':AMtransport[1], 
'Use':AMuse[1], 'Disposal':AMdisposal[1]}

        EnergyBarsAM["Total"] = sum(EnergyBarsAM.values())
        CO2BarsAM["Total"] = sum(CO2BarsAM.values())

    elif boundary == 'Cradle to Gate':
        EnergyBarsTM  = {'Material Production':TMmat_prod[0], 
'Manufacturing':TMmanuf[0]}
        CO2BarsTM = {'Material Production':TMmat_prod[1], 
'Manufacturing':TMmanuf[1]}

        EnergyBarsTM["Total"] = sum(EnergyBarsTM.values())
        CO2BarsTM["Total"] = sum(CO2BarsTM.values())
```

```
        EnergyBarsAM  = {'Material Production':AMmat_prod[0],
'Manufacturing':AMmanuf[0]}
        CO2BarsAM = {'Material Production':AMmat_prod[1],
'Manufacturing':AMmanuf[1]}

        EnergyBarsAM["Total"] = sum(EnergyBarsAM.values())
        CO2BarsAM["Total"] = sum(CO2BarsAM.values())

    else:
        EnergyBarsTM  = {'Manufacturing':TMmanuf[0]}
        CO2BarsTM = {'Manufacturing':TMmanuf[1]}

        EnergyBarsTM["Total"] = sum(EnergyBarsTM.values())
        CO2BarsTM["Total"] = sum(CO2BarsTM.values())

        EnergyBarsAM  = {'Manufacturing':AMmanuf[0]}
        CO2BarsAM = {'Manufacturing':AMmanuf[1]}

        EnergyBarsAM["Total"] = sum(EnergyBarsAM.values())
        CO2BarsAM["Total"] = sum(CO2BarsAM.values())

    return (EnergyBarsTM,CO2BarsTM,EnergyBarsAM,CO2BarsAM)
```

Completed - Main function of the program

Organizes graph displays, connected with ipywidgets, runs solver and main simulation

```
def Completed(imassTM = 5.13,
              fmassTM=2.04,
              embTMe=668,
              embTMc=39.5,
              embAMe=723,
              embAMc=40,
              embERecTM=500,
              embCRecTM=500,
              embERecAM=500,
              embCRecAM=500,
              embEDisTM=500,
              embCDisTM=500,
              embEDisAM=500,
              embCDisAM=500,
              recAM=0.15,
              recTM=0.15,
              TM11 = 2.28,
              TM12 = 0.17,
              TM13 = 0.85,
              TM21 = 18.5,
              TM22 = 1.53,
              TM23 = 0.15,
```

```
            TM31 = 0,
            TM32 = 0,
            TM33 = 0,
            TM41 = 0,
            TM42 = 0,
            TM43 = 0,
            TM51 = 0,
            TM52 = 0,
            TM53 = 0,
            AM11 = 176.5,
            AM12 = 9.6,
            AM13 = 1,
            AM21 = 18.5,
            AM22 = 1.39,
            AM23 = 0.1,
            AM31 = 0,
            AM32 = 0,
            AM33 = 0,
            AM41 = 0,
            AM42 = 0,
            AM43 = 0,
            AM51 = 0,
            AM52 = 0,
            AM53 = 0,
            TMsteps=2,
            AMsteps=2,
            secondary = True,
            mass_reduction_perc=0,
            finishing_perc=0.1,
            support_perc = 0.2,
            eff_energ_conv = 0.4,
            CO2equivalent = 20,
            transpTM = 'Ocean Shipping - Diesel',
            transpAM = 'Ocean Shipping - Diesel',
            distTM = 300,
            distAM = 5,
            useType = 'Long-Distance Aircraft',
            boundary='Cradle to Grave',
            solver = True,
            gType = "All"):
    '''
    Description:
    ------------

    This function calls the Calculation function and is called by the 
ipywidgets
    library to create the control interface for the program

    Variables:
    ----------
```

```
    imassTM:
    - Type: float
    - Desc: Stores the initial mass of the TM workpiece
    fmassTM:
    - Type: float
    - Desc: Stores the final mass after the workpiece has been
finished in TM
    embTMe:
    - Type: float
    - Desc: Stores the embodied energy impact for the material per
unit of material
    used in TM
    embTMc:
    - Type: float
    - Desc: Stores the embodied CO2 impact for the material per unit
of material
    used in TM
    embAMe:
    - Type: float
    - Desc: Stores the embodied energy impact for the material per
unit of material
    used in AM
    embAMc:
    - Type: float
    - Desc: Stores the embodied CO2 impact for the material per unit
of material
    used in AM
    embEDisTM:
    - Type: Float
    - Desc: Stores the embodied energy for disposal of material per
unit of material used in TM
    embCDisTM:
    - Type: Float
    - Desc: Stores the embodied carbon for disposal of material per
unit of material used in TM
    embEDisAM:
    - Type: Float
    - Desc: Stores the embodied energy for disposal of material per
unit of material used in AM
    embCDisAM:
    - Type: Float
    - Desc: Stores the embodied carbon for disposal of material per
unit of material used in AM
    embERecTM:
    - Type: Float
    - Desc: Stores the embodied energy impact from recycled material
input per unit of material used in TM
    embCRecTM:
    - Type: Float
```

```
    - Desc: Stores the embodied carbon from recycled material input
per unit of material used in TM
    embERecAM:
    - Type: Float
    - Desc: Stores the embodied energy impact from recycled material
input per unit of material used in AM
    embCRecAM:
    - Type: Float
    - Desc: Stores the embodied carbon from recycled material input
per unit of material used in AM
    TM11:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
first step
    of the TM path
    TM12:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
first step
    of TM path
    TM13:
    - Type: float
    - Desc: Stores the mass that will be removed in first step of the
TM path
    TM21:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
second step
    of the TM path
    TM22:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
second step
    of TM path
    TM23:
    - Type: float
    - Desc: Stores the mass that will be removed in second step of the
TM path
    TM31:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
third step
    of the TM path
    TM32:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
third step
    of TM path
    TM33:
    - Type: float
```

```
    - Desc: Stores the mass that will be removed in third step of the
TM path
    TM41:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
fourth step
    of the TM path
    TM42:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
fourth step
    of TM path
    TM43:
    - Type: float
    - Desc: Stores the mass that will be removed in fourth step of the
TM path
    TM51:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
fifth step
    of the TM path
    TM52:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
fifth step
    of TM path
    TM53:
    - Type: float
    - Desc: Stores the mass that will be removed in fifth step of the
TM path
    TMsteps:
    - Type: float
    - Desc: Stores the ammount of steps that shpuld be considerd in
this
    simulation for the TM path, as in if the number 3 is selected 3
    steps will be considered, if 2 is selected only the first two
steps will
    be considered. Can go up to 5
    AM11:
    - Type: float
    - Desc: Stores the energy usage per unit of material
removed/placed in first step
    of the AM path
    AM12:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
first step
    of AM path
    AM13:
    - Type: float
```

```
    - Desc: Stores the mass that will be removed/placed in first step
of the AM path
    AM21:
    - Type: float
    - Desc: Stores the energy usage per unit of material
removed/placed in second step
    of the AM path
    AM22:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
second step
    of AM path
    AM23:
    - Type: float
    - Desc: Stores the mass that will be removed/placed in second step
of the AM path
    AM31:
    - Type: float
    - Desc: Stores the energy usage per unit of material
removed/placed in third step
    of the AM path
    AM32:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
third step
    of AM path
    AM33:
    - Type: float
    - Desc: Stores the mass that will be removed/placed in third step
of the AM path
    AM41:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
fourth step
    of the AM path
    AM42:
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
fourth step
    of AM path
    AM43:
    - Type: float
    - Desc: Stores the mass that will be removed in fourth step of the
AM path
    TM51:
    - Type: float
    - Desc: Stores the energy usage per unit of material removed in
fifth step
    of the AM path
    AM52:
```

```
    - Type: float
    - Desc: Stores the CO2 emission per unit of material removed in
fifth step
    of AM path
    AM53:
    - Type: float
    - Desc: Stores the mass that will be removed in fifth step of the
AM path
    AMsteps:
    - Type: float
    - Desc: Stores the ammount of steps that shpuld be considerd in
this
    simulation for the AM path, as in if the number 3 is selected all
3
    steps will be considered, if 2 is selected only the first two
steps will
    be considered
    mass_reduction_perc:
    - Type: float
    - Desc: Percentage of the final mass of the product produced
through TM,
    that will be reduced due to geometry optimization in AM path
    finishing_perc:
    - Type: float
    - Desc: Percentage of additional mass that is placed in the AM
process due
    to machine allowance that will later need to be removed
    support_perc:
    - Type: float
    - Desc: Percentage of additional mass that is placed in the AM
process due
    to the need to create supports during the build of a particular
piece
    eff_energ_conv:
    - Type: float
    - Desc: Stores the efficiency of the Power Grid considered in the
simulation
    CO2equivalent:
    - Type: float
    - Desc: Stores the equivalent CO2 emissions due to the energy
required to
    produce the piece in the considered Power Grid in the simulation


    recTM:
    - Type: float
    - Desc: % of recyclability (combined recovery, recycling rate and
recycling mateiral yield)
```

```
    recAM:
    - Type: float
    - Desc: % of recyclability (combined recovery, recycling rate and
recycling mateiral yield)

    secondary:
    - Type: boolean
    - Desc: Serves as a flag of whether or no the input for
manufacturing steps is in primary or secondary use

    useType:
    - Type: string
    - Desc: Stores the input of where the user wants to consider
thepiece to be used in;

    transpTM:
    - Type: string
    - Desc: Stores the name of the transportation mode that will be
used for TM, serves as a key for
    the dictionary of specific impacts

    transpAM:
    - Type: string
    - Desc: Stores the name of the transportation mode that will be
used for AM, serves as a key for
    the dictionary of specific impacts

    distTM:
    - Type: float
    - Desc:  Stores the distance that will be travelled between
manufacturing and
    use phase in TM path

    distAM:
    - Type: float
    - Desc: Stores the distance that will be travelled between
manufacturing and
    use phase in AM path

    boundary:
    - Type: String
    - Desc: Stores boundary that will be considered for this
simulation, depending
    on its value graphs will display either a Cradle to grave, or
Cradle to Gate or Gate to Gate scenario

    solver:
    - Type: string
    - Desc: Serves as a flag from the user to indicate whether the
breakeven percentage
```

```
    of mass reduction from TM to AM should be calculated;

    gType:
    - Type: string
    - Desc: Serves as an input from the user to decide the type of
graph they want to see



    Returns:
    --------

    Doesn't return anything but print out the desided plot

    '''
    # Calls Calculation function to receive valiues for each phase
    EnergyBarsTM,CO2BarsTM,EnergyBarsAM,CO2BarsAM =
Calculation(imassTM,

fmassTM,

embTMe,

embTMc,

embAMe,

embAMc,

embERecTM,

embCRecTM,

embERecAM,

embCRecAM,

embEDisTM,

embCDisTM,

embEDisAM,

embCDisAM,
                                                                  recAM,

                                                                  recTM,
                                                                  TM11 ,
```

```
TM12 ,
TM13 ,
TM21 ,
TM22 ,
TM23,
TM31,
TM32,
TM33,
TM41,
TM42,
TM43,
TM51,
TM52,
TM53,
AM11,
AM12,
AM13,
AM21,
AM22,
AM23,
AM31,
AM32,
AM33,
AM41,
AM42,
AM43,
AM51,
AM52,
AM53,

TMsteps,

AMsteps,

secondary,

mass_reduction_perc,

finishing_perc,

support_perc,

eff_energ_conv,

CO2equivalent,

transpTM ,

transpAM ,
```

```
                                                                  distTM
,
                                                                  distAM
,

useType,

boundary)



    ## INITIALIZING SOLVER TO FIND EQUILIBRIUM OF ENERGY IN FUCNTION
MASS REDUCTION
    mass_reduc_solver=0.1
    step = 0.1
    itera = 0
    last_result = 0
    # Initializes a loop that basically runs a Newton Method to find
the correct value
    while solver and (itera<50):
        EnergyBarsTMs,CO2BarsTMs,EnergyBarsAMs,CO2BarsAMs =
Calculation(imassTM,

fmassTM,

embTMe,

embTMc,

embAMe,

embAMc,

embERecTM,

embCRecTM,

embERecAM,

embCRecAM,

embEDisTM,

embCDisTM,

embEDisAM,

embCDisAM,

recAM,
```

```
recTM,
TM11 ,
TM12 ,
TM13 ,
TM21 ,
TM22 ,
TM23,
TM31,
TM32,
TM33,
TM41,
TM42,
TM43,
TM51,
TM52,
TM53,
AM11,
AM12,
AM13,
AM21,
AM22,
AM23,
AM31,
AM32,
AM33,
```

```
AM41,

AM42,

AM43,

AM51,

AM52,

AM53,

TMsteps,

AMsteps,

secondary,

mass_reduc_solver,

finishing_perc,

support_perc,

eff_energ_conv,

CO2equivalent,

transpTM ,

transpAM ,

distTM ,

distAM ,

useType,

boundary)

        itera += 1
        if (EnergyBarsTMs["Total"] - EnergyBarsAMs["Total"] > 0) :
            result = -1
        else:
            result = +1

        if itera == 1:
            last_result = result
```

```python
        if last_result != result:
            step = step/2

        mass_reduc_solver = mass_reduc_solver + result*step
        last_result = result

    # prints result from the solver
    print("Equilibrium of reduction in mass: ",mass_reduc_solver)


    ### START display code
    # Degree of rotation in xlabels
    rotation_deg = 30

    # Selects graphs according to user input, in this scenario
    # shows graphs for Energy and CO2 for both TM and AM
    if gType == "All":

        fig, ((ax1,ax2),(ax3,ax4)) = plt.subplots(2,2, sharey='row')

        namesE = list(EnergyBarsTM.keys())
        valuesE = list(EnergyBarsTM.values())

        ax1.bar(namesE,valuesE, color='red')
        ax1.set_title('Traditional Manufacturing')
        ax1.set_ylabel('Energy Consumption (MJ)')
        ax1.tick_params(axis='x',labelsize=15,
labelrotation=rotation_deg)

        namesC = list(CO2BarsTM.keys())
        valuesC = list(CO2BarsTM.values())

        ax3.bar(namesC,valuesC, color='red')
        ax3.set_ylabel('CO2 production (kg)')
        ax3.tick_params(axis='x',labelsize=15,
labelrotation=rotation_deg)

        namesE = list(EnergyBarsAM.keys())
        valuesE = list(EnergyBarsAM.values())

        ax2.bar(namesE,valuesE, color='blue')
        ax2.set_title('Additive Manufacturing')
        ax2.set_ylabel('Energy Consumption (MJ)')
        ax2.tick_params(axis='x',labelsize=15,
labelrotation=rotation_deg)

        namesC = list(CO2BarsAM.keys())
        valuesC = list(CO2BarsAM.values())
```

```
        ax4.bar(namesC,valuesC, color='blue')
        ax4.set_ylabel('CO2 production (kg)')
        ax4.tick_params(axis='x',labelsize=15,
labelrotation=rotation_deg)

        fig.set_size_inches(18.5, 15)

    # Shows results as the delta between TM and Am for each phase
    else:
        namesE = list(EnergyBarsTM.keys())
        valuesE = np.subtract(list(EnergyBarsTM.values()),
list(EnergyBarsAM.values()))

        namesC = list(CO2BarsTM.keys())
        valuesC = np.subtract(list(CO2BarsTM.values()),
list(CO2BarsAM.values()))

        fig, (ax1,ax2) = plt.subplots(1,2)
        ax1.bar(namesE,valuesE, color='red')
        ax1.set_title('Difference in Energy Consumption')
        ax1.set_ylabel('Delta in Energy Consumption (MJ)')
        ax1.tick_params(axis='x',labelsize=15,
labelrotation=rotation_deg)


        ax2.bar(namesC,valuesC, color='green')
        ax2.set_title('Difference in CO2 Consumption')
        ax2.set_ylabel('Delta in CO2 production (kg)')
        ax2.tick_params(axis='x',labelsize=15,
labelrotation=rotation_deg)

        fig.set_size_inches(18.5, 10.5)


    plt.rcParams.update({'font.size': 15})

    fig.tight_layout()
    plt.show(fig)

    # Prints results achieved for each phase
    print("Energy:\n")
    print("TM\n",EnergyBarsTM)
    print("AM\n",EnergyBarsAM)
    print("Carbon:\n")
    print("TM\n",CO2BarsTM)
    print("AM\n",CO2BarsAM)
```

Read Input

Very useful function ofr usability, not necessary for the program exactly but very welcome in practicality

```
def ReadInput(file_name):
    '''
    Description:
    ------------

    This function is designed to read a .txt file and extract the 
inputs to use in the simulation,
    it is made to simply inputting numbers if you want to reproduce a 
previous simulation

    Variables:
    ----------

    file_name:
    - Type: string
    - Desc: Stores the name of the file from which the inputs will be 
extracted



    Returns:
    --------

    inputDict:
    - Type: Dictionary
    - Desc: Dictionary of inputs, each key is the name of the variable 
and
    the values are the input values

    '''
    # Initializes dictionary
    inputDict = {}
    # Opens desired file
    f = open(file_name,'r')

    # Loop to go thorugh each line one by one
    for line in f.readlines():

        # Separating variable name from value to add entry to 
dictionary
        words = line.split('=')
        key = words[0]
        value = words[1].replace('\
n','').replace('"','').replace("'",'')
```

```
        # Tries to convert string into float, in case there is an
error,
        # it means that the variable type was supposed to be string as
it alredy is and doesn't need any conversion
        try:
            value = float(value)
        except:
            pass

        inputDict[key]=value
    return inputDict

# Calls function to store initial input
I = ReadInput("input09-02-2023--18-30-30")
print(I)
```

```
{'imassTM': 1.515, 'fmassTM': 1.378, 'embTMe': 32.0, 'embTMc': 2.4,
'embAMe': 190.0, 'embAMc': 12.0, 'embERecTM': 8.5, 'embCRecTM': 0.6,
'embERecAM': 32.0, 'embCRecAM': 2.55, 'embEDisTM': 0.0, 'embCDisTM':
0.0, 'embEDisAM': 0.0, 'embCDisAM': 0.0, 'recAM': 0.9, 'recTM': 0.85,
'TM11': 10.0, 'TM12': 0.7, 'TM13': 1.0, 'TM21': 5.0, 'TM22': 0.4,
'TM23': 0.1, 'TM31': 0.0, 'TM32': 0.0, 'TM33': 0.0, 'TM41': 0.0,
'TM42': 0.0, 'TM43': 0.0, 'TM51': 0.0, 'TM52': 0.0, 'TM53': 0.0,
'AM11': 328.0, 'AM12': 9.6, 'AM13': 1.0, 'AM21': 18.5, 'AM22': 1.39,
'AM23': 0.1, 'AM31': 0.0, 'AM32': 0.0, 'AM33': 0.0, 'AM41': 0.0,
'AM42': 0.0, 'AM43': 0.0, 'AM51': 0.0, 'AM52': 0.0, 'AM53': 0.0,
'TMsteps': 2.0, 'AMsteps': 1.0, 'secondary': 'True',
'mass_reduction_perc': 0.7, 'finishing_perc': 0.0, 'support_perc':
0.2, 'eff_energ_conv': 0.4, 'CO2equivalent': 0.0711, 'transpTM':
'Ocean Shipping - Diesel', 'transpAM': 'Ocean Shipping - Diesel',
'distTM': 0.0, 'distAM': 0.0, 'useType': 'Passenger Car - Diesel',
'boundary': 'Cradle to Grave', 'solver': 'True', 'gType': 'All'}
```

INTERACTIVE PROGRAM

```
# Initializes ipywidgets to begin creating interface for the program
# Each input type is defined according to library references, value is
extracted from 'I'
# dictionary which stores input from selected file
widget = ipywidgets.interactive(Completed,
            solver = True,
            useType = ipywidgets.Dropdown(options=['Long-Distance
Aircraft','Short-Distance Aircraft','Articulated Truck','Passenger Car
- Diesel', 'Passenger Car -
Gasoline'],value=I["useType"],disabled=False),
            gType =
ipywidgets.Dropdown(options=['All','Difference'],value=I["gType"],disa
bled=False),
            boundary = ipywidgets.Dropdown(options=['Cradle to
Grave','Cradle to Gate','Gate to
Gate'],value=I["boundary"],disabled=False),
```

```
            transpTM = ipywidgets.Dropdown(options=['Ocean Shipping -
Diesel','Coastal Shipping - Diesel','Barge - Diesel','Rail - Diesel',
'Articulated HGV (up to 55 metric tons) - Diesel', '40 metric ton
truck - Diesel', '32 metric ton truck - Diesel', '14 metric ton truck
- Diesel', 'Light goods vehicle - Diesel', 'Family car - Diesel',
'Family car - Gasoline', 'Family car -LPG', 'Family car - Hybrid
Gasoline-electric', 'Super sports car and SUV - Gasoline', 'Long Haul
Aircraft - Kerosene', 'Short Haul Aircraft - Kerosene', 'Helicopter
(Eurocopter AS350) - Kerosene'],value=I["transpTM"],disabled=False),
            transpAM = ipywidgets.Dropdown(options=['Ocean Shipping -
Diesel','Coastal Shipping - Diesel','Barge - Diesel','Rail - Diesel',
'Articulated HGV (up to 55 metric tons) - Diesel', '40 metric ton
truck - Diesel', '32 metric ton truck - Diesel', '14 metric ton truck
- Diesel', 'Light goods vehicle - Diesel', 'Family car - Diesel',
'Family car - Gasoline', 'Family car -LPG', 'Family car - Hybrid
Gasoline-electric', 'Super sports car and SUV - Gasoline', 'Long Haul
Aircraft - Kerosene', 'Short Haul Aircraft - Kerosene', 'Helicopter
(Eurocopter AS350) - Kerosene'],value=I["transpAM"],disabled=False),
            distTM = ipywidgets.FloatText(value=I["distTM"]),
            distAM = ipywidgets.FloatText(value=I["distAM"]),

            imassTM =ipywidgets.FloatText(value=I["imassTM"]) ,
            fmassTM =ipywidgets.FloatText(value=I["fmassTM"]) ,
            TM11 = ipywidgets.FloatText(value=I["TM11"]),
            TM12 = ipywidgets.FloatText(value=I["TM12"]),
            TM13 =
ipywidgets.FloatSlider(value=I["TM13"],min=0,max=1,step=0.05,disabled=
False,continuous_update=False,readout=True),
            TM21 = ipywidgets.FloatText(value=I["TM21"]),
            TM22 = ipywidgets.FloatText(value=I["TM22"]),
            TM23 =
ipywidgets.FloatSlider(value=I["TM23"],min=0,max=1,step=0.05,disabled=
False,continuous_update=False,readout=True),
            TM31 = ipywidgets.FloatText(value=I["TM31"]),
            TM32 = ipywidgets.FloatText(value=I["TM32"]),
            TM33 =
ipywidgets.FloatSlider(value=I["TM33"],min=0,max=1,step=0.05,disabled=
False,continuous_update=False,readout=True),
            TM41 = ipywidgets.FloatText(value=I["TM41"]),
            TM42 = ipywidgets.FloatText(value=I["TM42"]),
            TM43 =
ipywidgets.FloatSlider(value=I["TM43"],min=0,max=1,step=0.05,disabled=
False,continuous_update=False,readout=True),
            TM51 = ipywidgets.FloatText(value=I["TM51"]),
            TM52 = ipywidgets.FloatText(value=I["TM52"]),
            TM53 =
ipywidgets.FloatSlider(value=I["TM53"],min=0,max=1,step=0.05,disabled=
False,continuous_update=False,readout=True),
            TMsteps =
ipywidgets.IntSlider(value=I["TMsteps"],min=1,max=5,step=1,disabled=Fa
```

```
lse,continuous_update=False,readout=True),
            AM11 = ipywidgets.FloatText(value=I["AM11"]),
            AM12 = ipywidgets.FloatText(value=I["AM12"]),
            AM13 =
ipywidgets.FloatSlider(value=I["AM13"],min=0,max=1,step=0.05,disabled=
False,continuous_update=False,readout=True),
            AM21 = ipywidgets.FloatText(value=I["AM21"]),
            AM22 = ipywidgets.FloatText(value=I["AM22"]),
            AM23 =
ipywidgets.FloatSlider(value=I["AM23"],min=0,max=1,step=0.05,disabled=
False,continuous_update=False,readout=True),
            AM31 = ipywidgets.FloatText(value=I["AM31"]),
            AM32 = ipywidgets.FloatText(value=I["AM32"]),
            AM33 =
ipywidgets.FloatSlider(value=I["AM33"],min=0,max=1,step=0.05,disabled=
False,continuous_update=False,readout=True),
            AM41 = ipywidgets.FloatText(value=I["AM41"]),
            AM42 = ipywidgets.FloatText(value=I["AM42"]),
            AM43 =
ipywidgets.FloatSlider(value=I["AM43"],min=0,max=1,step=0.05,disabled=
False,continuous_update=False,readout=True),
            AM51 = ipywidgets.FloatText(value=I["AM51"]),
            AM52 = ipywidgets.FloatText(value=I["AM52"]),
            AM53 =
ipywidgets.FloatSlider(value=I["AM53"],min=0,max=1,step=0.05,disabled=
False,continuous_update=False,readout=True),

AMsteps=ipywidgets.IntSlider(value=I["AMsteps"],min=1,max=5,step=1,dis
abled=False,continuous_update=False,readout=True),

mass_reduction_perc=ipywidgets.FloatSlider(value=I["mass_reduction_per
c"],min=0,max=1,step=0.05,disabled=False,continuous_update=False,reado
ut=True),

finishing_perc=ipywidgets.FloatSlider(value=I["finishing_perc"],min=0,
max=1,step=0.05,disabled=False,continuous_update=False,readout=True),
            support_perc =
ipywidgets.FloatSlider(value=I["support_perc"],min=0,max=1,step=0.05,d
isabled=False,continuous_update=False,readout=True),
            eff_energ_conv =
ipywidgets.FloatSlider(value=I["eff_energ_conv"],min=0,max=1,step=0.05
,disabled=False,continuous_update=False,readout=True),
            CO2equivalent =
ipywidgets.FloatText(value=I["CO2equivalent"]),

recAM=ipywidgets.FloatSlider(value=I["recAM"],min=0,max=1,step=0.05,di
sabled=False,continuous_update=False,readout=True),

recTM=ipywidgets.FloatSlider(value=I["recTM"],min=0,max=1,step=0.05,di
sabled=False,continuous_update=False,readout=True),
```

```
            embTMe=ipywidgets.FloatText(value=I["embTMe"]),
            embTMc=ipywidgets.FloatText(value=I["embTMc"]),
            embAMe=ipywidgets.FloatText(value=I["embAMe"]),
            embAMc=ipywidgets.FloatText(value=I["embAMc"]),
            embERecTM=ipywidgets.FloatText(value=I["embERecTM"]),
            embCRecTM=ipywidgets.FloatText(value=I["embCRecTM"]),
            embERecAM=ipywidgets.FloatText(value=I["embERecAM"]),
            embCRecAM=ipywidgets.FloatText(value=I["embCRecAM"]),
            embEDisTM=ipywidgets.FloatText(value=I["embEDisTM"]),
            embCDisTM=ipywidgets.FloatText(value=I["embCDisTM"]),
            embEDisAM=ipywidgets.FloatText(value=I["embEDisAM"]),
            embCDisAM=ipywidgets.FloatText(value=I["embCDisAM"]))


# controls = HBox(widget.children[:-1], layout = Layout(flex_flow='row
wrap'))

# Separates widgets into groups, so that they can be displayed
together
mass = HBox(widget.children[0:2], layout = Layout(flex_flow='row
wrap'))
materialProperties = HBox(widget.children[2:16], layout =
Layout(flex_flow='row wrap'))
manufacturing = HBox(widget.children[16:49], layout =
Layout(flex_flow='row wrap'))
percentages = HBox(widget.children[49:52], layout =
Layout(flex_flow='row wrap'))
powerGrid = HBox(widget.children[52:54], layout =
Layout(flex_flow='row wrap'))
transport1 = HBox(widget.children[54:56], layout =
Layout(flex_flow='row wrap'))
transport2 = HBox(widget.children[56:58], layout =
Layout(flex_flow='row wrap'))
control = HBox(widget.children[58:62], layout = Layout(flex_flow='row
wrap'))

output = widget.children[-1]

# Creates button that will be used to save present input
button = ipywidgets.Button(description="Save input")

# Displays widgets and button and output
display(VBox([mass,materialProperties,manufacturing,percentages,powerG
rid,transport1,transport2,control, output, button]))

# Defining button function that receives all widgets and saves their
values to a file whose name depends on the present date and time,
# in order to avoid file names with the same name
def on_button_clicked(b,children=[]):
```

```
    title = "input"+date.now().strftime("%d-%m-%Y--%H-%M-%S")
    f= open(title,"w+")
    for i in range(len(children)-1):
         f.write(children[i].description + "=" +
str(children[i].value)+"\n")
    f.close()
    with output:
        print("Input Saved as " + title+".")

button.on_click(functools.partial(on_button_clicked,
children=widget.children))
```

{"model_id":"897101e5f8ef4d848b8459df4e987639","version_major":2,"version_minor":0}

```
conta = 0
for i in widget.children:

    print(conta,i.description)
    conta += 1
```

```
0 imassTM
1 fmassTM
2 embTMe
3 embTMc
4 embAMe
5 embAMc
6 embERecTM
7 embCRecTM
8 embERecAM
9 embCRecAM
10 embEDisTM
11 embCDisTM
12 embEDisAM
13 embCDisAM
14 recAM
15 recTM
16 TM11
17 TM12
18 TM13
19 TM21
20 TM22
21 TM23
22 TM31
23 TM32
24 TM33
25 TM41
26 TM42
27 TM43
28 TM51
```

```
29 TM52
30 TM53
31 AM11
32 AM12
33 AM13
34 AM21
35 AM22
36 AM23
37 AM31
38 AM32
39 AM33
40 AM41
41 AM42
42 AM43
43 AM51
44 AM52
45 AM53
46 TMsteps
47 AMsteps
48 secondary
49 mass_reduction_perc
50 finishing_perc
51 support_perc
52 eff_energ_conv
53 CO2equivalent
54 transpTM
55 transpAM
56 distTM
57 distAM
58 useType
59 boundry
60 solver
61 gType

----------------------------------------------------------------------
-----
AttributeError                            Traceback (most recent call
last)
<ipython-input-264-871549ce0234> in <module>
      2 for i in widget.children:
      3 
----> 4     print(conta,i.description)
      5     conta += 1

AttributeError: 'Output' object has no attribute 'description'
```

```
Calculation(imassTM = 5.13,
              fmassTM=2.04,
              embTMe=668,
              embTMc=39.5,
```

```
embAMe=723,
embAMc=40,
embERecTM=500,
embCRecTM=500,
embERecAM=500,
embCRecAM=500,
embEDisTM=500,
embCDisTM=500,
embEDisAM=500,
embCDisAM=500,
recAM=0.15,
recTM=0.15,
TM11 = 2.28,
TM12 = 0.17,
TM13 = 0.85,
TM21 = 18.5,
TM22 = 1.53,
TM23 = 0.15,
TM31 = 0,
TM32 = 0,
TM33 = 0,
TM41 = 0,
TM42 = 0,
TM43 = 0,
TM51 = 0,
TM52 = 0,
TM53 = 0,
AM11 = 176.5,
AM12 = 9.6,
AM13 = 1,
AM21 = 18.5,
AM22 = 1.39,
AM23 = 0.1,
AM31 = 0,
AM32 = 0,
AM33 = 0,
AM41 = 0,
AM42 = 0,
AM43 = 0,
AM51 = 0,
AM52 = 0,
AM53 = 0,
TMsteps=2,
AMsteps=2,
secondary = True,
mass_reduction_perc=0,
finishing_perc=0.1,
support_perc = 0.2,
eff_energ_conv = 0.4,
CO2equivalent = 20,
```

```
transpTM = 'Ocean Shipping - Diesel',
transpAM = 'Ocean Shipping - Diesel',
distTM = 300,
distAM = 5,
useType = 'Long-Distance Aircraft',
boundry='Cradle to Grave')
```