Progress in Automated System Design by Evolutionary Algorithms

IEPC-2019-A-456

Presented at the 36th International Electric Propulsion Conference
University of Vienna, Austria
September 15-20, 2019

Manfred Ehresmann*
University of Stuttgart, Stuttgart, Baden-Württemberg, 70569, Germany

Themistoklis Spanoudis†
Aristotle University of Thessaloniki, Thessaloniki, 54124, Greece

Jonathan Skalden‡
University of Stuttgart, Stuttgart, Baden-Württemberg, 70569, Germany

Georg Herdrich§
University of Stuttgart, Stuttgart, Baden-Württemberg, 70569, Germany

Stefanos Fasoulas¶
University of Stuttgart, Stuttgart, Baden-Württemberg, 70569, Germany

This paper presents improvements and updates to the Evolutionary System Design Converger (ESDC). ESDC is a software tool for system and mission analysis, which allows to estimate (sub-)system masses as well as performance parameters for specified missions. To solve this multidimensional optimization problem, an evolutionary algorithm is utilized that attempts to minimize the total system mass. Input interfaces have been simplified and extended to provide a machine useable interface. Scaling of performance and physical hardware is now fully generic and is self-updated from a provided database. This database has been further extended by publicly available data. The software tool has been adapted to operate fully open-source, with Octave 4.4.1 and above. The new visualisation module of the ESDC allows for fast analysis of evolutionary data in n dimensions for multiple lineages during their full evolutionary history. The ESDC is operating on a dedicated server as part of the Digital Concurrent Engineering Platform (DCEP) of a research programme named "Integrated Research Platform of Affordable Satellite" (IRAS), where it provides systems engineering solutions currently focused currently on electric propulsion system data.

Nomenclature

c_e = Effective exhaust velocity / m/s
m = Mass / kg
P = Power / W
Δv = Velocity increment / m/s

*Phd Student, Institute of Space Systems, ehresmann@irs.uni-stuttgart.de
†Google Summer of Code Scholarship Student, Mechanical Engineering, themistos@meng.auth.gr
‡Phd Student, Institute of Space Systems, skalden@irs.uni-stuttgart.de
§Head of plasma wind tunnel and electric propulsion, Institute of Space Systems, herdrich@irs.uni-stuttgart.de
¶Head of Institute, Institute of Space Systems, fasoulas@irs.uni-stuttgart.de

I. Introduction

This paper details the current capabilities of the Evolutionary System Design Converger (ESDC), describe the software tool design approach and applicability and currently producable outputs to demonstrate the functionality.

The ESDC is a holistic spacecraft design tool that uses a hardware data base containing physical and performance parameters to derive generic scaling laws for the realistic estimation of custom (sub)system component properties. An optimal spacecraft configuration is determined, when all input requirements are met and a minimal overall (system) mass is achieved. The system mass is viewed as a proxy for total system costs.

The code of the ESDC is open-source and a recent version is publicly available on the version-control platform GitHub [1] under a MIT license. The code was initially written in MATLAB[2], but has been overhauled and adapted to work with the free open-source alternative Octave, in version 4.4.1.[3].

As a hardware database can contain proprietary data, a basic database is provided as part of the repository and can be utilized by a user to be extended with proprietary data for an appropriate system design.

The ESDC is currently deployed on a server of the Institute of Space Systems (IRS), where it can be remotely called by a server or workstation of the German aerospace center (DLR) as an expert tool that is part of the Digital Concurrent Engineering Platform (DCEP)[4,5].

The concept of the DCEP envisions the integration of several remote expert tools via a DLR central node, which allows holders of knowledge (i.e. research and development entities) to supply results of their respective design tools upon request of a DCEP user automatically, without being forced to give up intellectual property. By this, a fast, streamlined and thus cost effective spacecraft design process shall be achieved, which is one of the research objectives of the project "Integrated Research Platform for Affordable Satellites" (IRAS).

To this end, a generic interface to allow data exchange has been implemented, which relies on the exchange of XML files, containing physical parameters of a current design, as well as commands for the simulation to be executed. The XML format has been chosen due to its ability to be integrated with most tools, programming languages and its independence from specific operating systems.

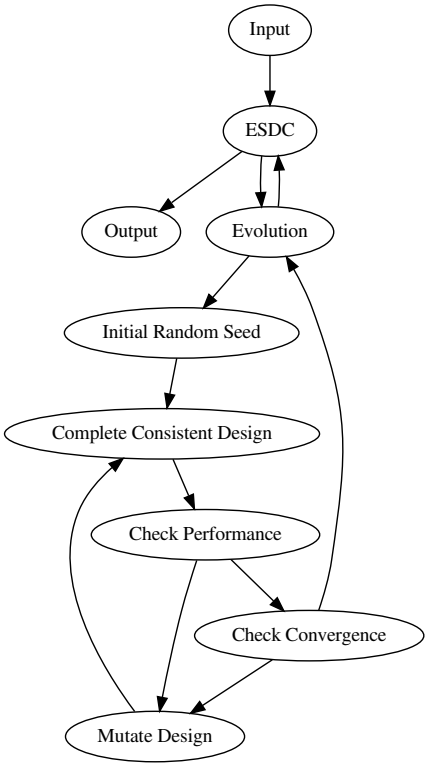

Figure 1. Evolutionary System Design Converger Overview Schematic

II. Evolutionary System Design Converger

The ESDC uses evolutionary algorithms to automatically design an optimal spacecraft system configuration. It allows for fast automated design iterations without subjective human influence. As main design driver, the modelling of an (electric) propulsion system has already been implemented, including the cross-relations to respective systems.
A basic software tool schematic of the ESDC working method is given in Fig. 1 and will be subsequently explained in further detail. Inputs containing simulation parameters, such as for example required velocity increment Δv, available electrical power P_0 or intended maximum system mass m_{max} is handed to the software tool.
The design tool then allows to automatically generate designs that fit these criteria and begins design iteration on the basis of a genetic algorithm, to determine an optimal system configuration. Once convergence conditions are met, the number of predefined optimal system configurations is given as output.

A. Inputs

In comparison to previous versions of the ESDC, the input interface has been significantly extended and is simplified for an inexperienced user, as well as easier to use when to be utilized as a digital interface when called from another computer.
The input interface current contains three XML-files: Mission parameters, simulation parameters and visualisation data, as given in Fig. 2. Only supplying mission parameters is a necessary input from a user for achieving a specific design. Here, for example, required velocity increment Δv, available electrical power P_0 or intended maximum system mass m_{max} are defined. The number of simultaneous design cases to be calculated is not limited. Previous versions of the ESDC required additional information on the propulsion system, propellant and other specific parameters. The new version is no longer limited by this. The previously required input cases (i.e. degrees of freedom) are now handled automatically internally. The number of input cases is not limited, allowing for convenient batch processing.
Simulation parameters define parameters that adapt internal constants that can directly influence the quality of reached design solutions, as well as the performance of the solver:

- Number of seed points / lineages to be mutated
- Maximum mutation jump range for continuous degrees of freedom
- Minimal mutation jump range for continuous degrees of freedom
- Number of failed mutations as convergence criteria
- XML output definition
- Verbosity to the Command Line Interface

Using provided default settings is highly recommended.
Additionally, a new module for visualizing data from produced solutions has been developed. This allows the generation of 2D and 3D graphs as well as animations for understanding the solutions generated by the tool. Respective documentation has been provided. Default settings, as well as numerous examples for appropriate definition are provided.
These user inputs are supplemented by a reference XML database that contains hardware information to be utilized for spacecraft hardware scaling.

B. Modelling

The ESDC is a software tool that uses evolutionary principles to determine optimal solutions in a multi-dimensional design space. Details on how to apply these to spacecraft design are briefly explained in the following.

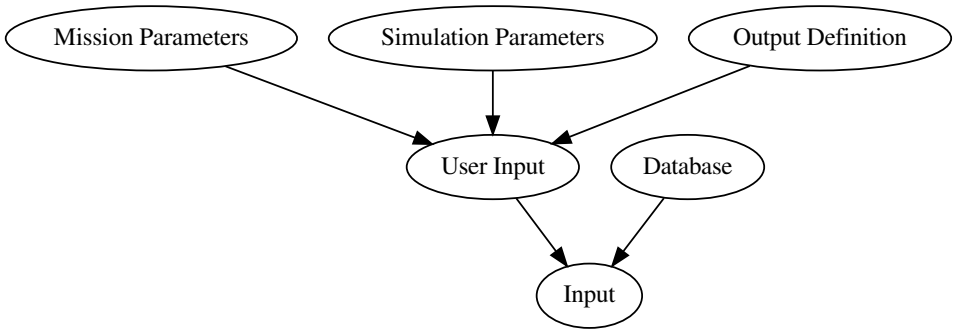

Figure 2. Evolutionary System Design Converger Input Interface Schematic

Any algorithm that combines (random) mutation and non-random selection can be described as evolutionary. The term genetic algorithm refers to the fact that information from a previous generation or design point is relayed to the next generation and mutated again, as illustrated in Fig. 3. By this, incremental improvement can be produced, when an appropriate selection mechanism is utilized.
As a main spacecraft design driver, the scaling of an electric propulsion system is currently implemented with related components, such as propellant, power processing units, power generation and tanks mass estimations.
The total electric propulsion subsystem mass is here utilized as objective performance indicator to determine best-suited designs. If the new design is incrementally advantageous, this new data (or generation/gene) is then utilized for further reiteration. If a preset number of non successful mutations is reached, it is assumed that convergence has been achieved.
Any design is directly influenced by user defined constraints (e.g. Δv) and requirements such as total manoeuvre burn time. The relevant relations to define a consistent electric propulsion system from mutated data are implemented.
It has to be noted that an evolutionary algorithm does not necessarily reach a global optimum. It is possible to only reach a local optimum within the design space, where every mutation is negative in relation to the already established design point. Thus, mitigation is necessary, which can be achieved with appropriate design space seeding.

C. Finding Global Optima

Using an evolutionary algorithm with a single lineage to be mutated for achieving a good solution is equivalent to a gamble, which may be successful by the coincidence of a lucky seed point. An evolutionary algorithm allows to trace a higher dimensional gradient and this path can end at the global optimum or being "trapped" at a local optimum. Two mitigation strategies are implemented for the ESDC.
First, an arbitrary number (i.e. user input) of seed points can be utilized. Each seed point generation considers first the given requirements for the spacecraft and then the degrees of freedom available in these designs. In the current version, degrees of freedom for the designs are predefined (e.g. maximum thrust, effective exhaust velocity, propulsion type, propellant and more). A future version will contain a generic degree of freedom determination method that will directly add degrees of freedom from additional database entries.
Second, a non-constant mutation range has been implemented. This means that for continuous degrees of freedom, for example effective exhaust velocity or thrust of the electric propulsion system, mutation is permitted within a predefined minimal range as well as within a permitted multiple of this change. By this, it is possible for the evolutionary algorithm to "jump" over gaps of a less ideal design space to a more ideal solution.

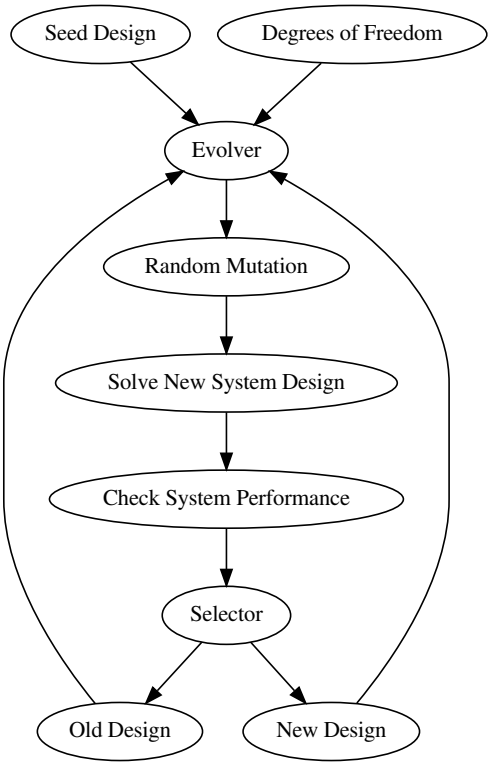

Figure 3. Evolutionary System Design Converger Evolution Schematic

D. Generic Scaling

A key advantage of the ESDC is the ability of generic scaling of components based on actual hardware data. In previous versions, direct scaling laws were manually derived from available public data.
This manual approach has now been upgraded to full generic scaling from an implemented database described in the following.
First, the database is accessed and a search will yield all types of hardware that are relevant in this step. Then, it is further checked whether all relevant fields of a piece of hardware are present for further processing. If not, the point is neglected. By this, a flexibility for XML input is now available, as now incomplete data might also be used in specific cases. The available data is then processed to be appropriate for scaling. For example, the calculation of the relation of electric propulsion thruster input power to thruster mass is generated.
This preprocessed data is then handed over to the interpolation system. In the current version, with limited numbers of database entries, this is handled by simple linear interpolation of nearest available points. Extrapolation is also implemented, but should be viewed as speculative estimation.
As this process does require accessing, searching and sorting data from a database, it is expected that a growing database will lead to significant performance decrease. Thus, mitigation in the form of lookup tables has been implemented. At a first cycle of the ESDC, a fresh table of interpolation data relevant for this specific case is generated. Thus, searching and sorting has to be performed just once. After that, only direct access to a system specific lookup table is required.
The generic scaling approach presented here is further relevant for the challenge of providing a generally available open-source software tool, while also many potential users have a strong interest in secrecy with no interest of publishing particular data. Thus, the ESDC repository contains base data, which can be expanded by a third-party by their private data without having to change the code directly.

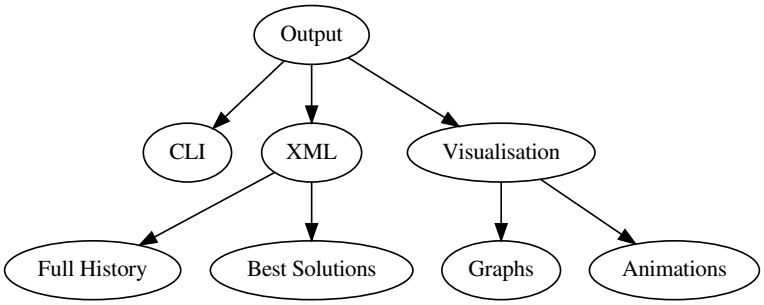

Figure 4. Evolutionary System Design Converger Output Schematic

E. Outputs

The ESDC is conceptualised for human and machine use alike. For this, the generated outputs are human and machine readable and are divided in three main sections, as illustrated in Fig. 4.

1. Command Line Interface

A command line interface (CLI) allows for directly observing the ESDC output while running, either within the Octave Guided User Interface (GUI), or GUI-less over a command line (i.e. shell) when connecting to a server. Here, basic states of an ongoing simulation are frequently displayed, like number of generations and how many lineages have converged, time until convergence, and more.

2. XML

For using the ESDC within a mission design tool chain, the XML output will likely be of greatest relevance, as any software that is capable of reading XML can use the data provided here to derive further conclusions and potentially start a ESDC simulation with iterated initial conditions. Two main XML output options are currently implemented. The first variant displays best solutions, up to an user specified number. In a naive case, only the single best solution would be desired. It might be the case that a very similar technological solution exists and a non-quantifiable reason for one or the other solution exists. Further, when multiple best candidates are displayed, a user can ascertain whether these solutions are nearly identical. Large differences might indicate insufficient convergence, minimal or non-existent differences indicate good convergence. This kind of output contains all relevant parameters like thruster parameters, estimated efficiencies, required power, masses of subsystems etc.
The second kind of XML output is relevant for code development and elaborates case studies. Here, the full evolutionary history of each lineage is produced, i.e. the full system output for each generation is produced. A respective analysis can be performed upon this data as it is often required during code development. This output should be disabled by default, as a comparatively large file needs to be generated in XML, requiring usually an additional minute of processing time on a conventional computer.

3. Visualisation

Mainly for better human understanding of the produced data, as well as to be better able to present the work of the ESDC, a dedicated module for automatically generating graphs from data has been implemented. This work was performed within the Google Summer of Code 2019 program[6].
Graphs displaying data of lineages per generation, multiple lineages, using custom dimensions to be displayed (i.e. x- dimension: c_e, y-dimension: thrust, z-dimension: masses) have been implemented for 2D and 3D plots, as still images and as animation.
The configuration of the visualization system is achieved through a user defined XML file. Any continuous degrees-of-freedom can be assigned on x- and y-axis on the 2D plots plus z-axis on the 3D plots. Additional

continuous or discrete degrees-of-freedom can be encoded on line style and line color, marker and marker color as well as on stacked bar graphs, allowing for visualization of multi-dimensional systems. Sorting and selecting lineages the to be visualized according to a chosen degree-of-freedom in increasing or decreasing order is also achievable through the XML interface. Another main features is the inclusion or not of failed mutations in the visualization as well as their corresponding appearance to assess the genetic algorithm more thoroughly and not only successful mutations.
The main purposes of these graphs is the simple evaluation and analysis of the evolutionary behaviour of a system to be optimized. General system design principles can be derived. In the following sections, examples of this visualisation system are presented.

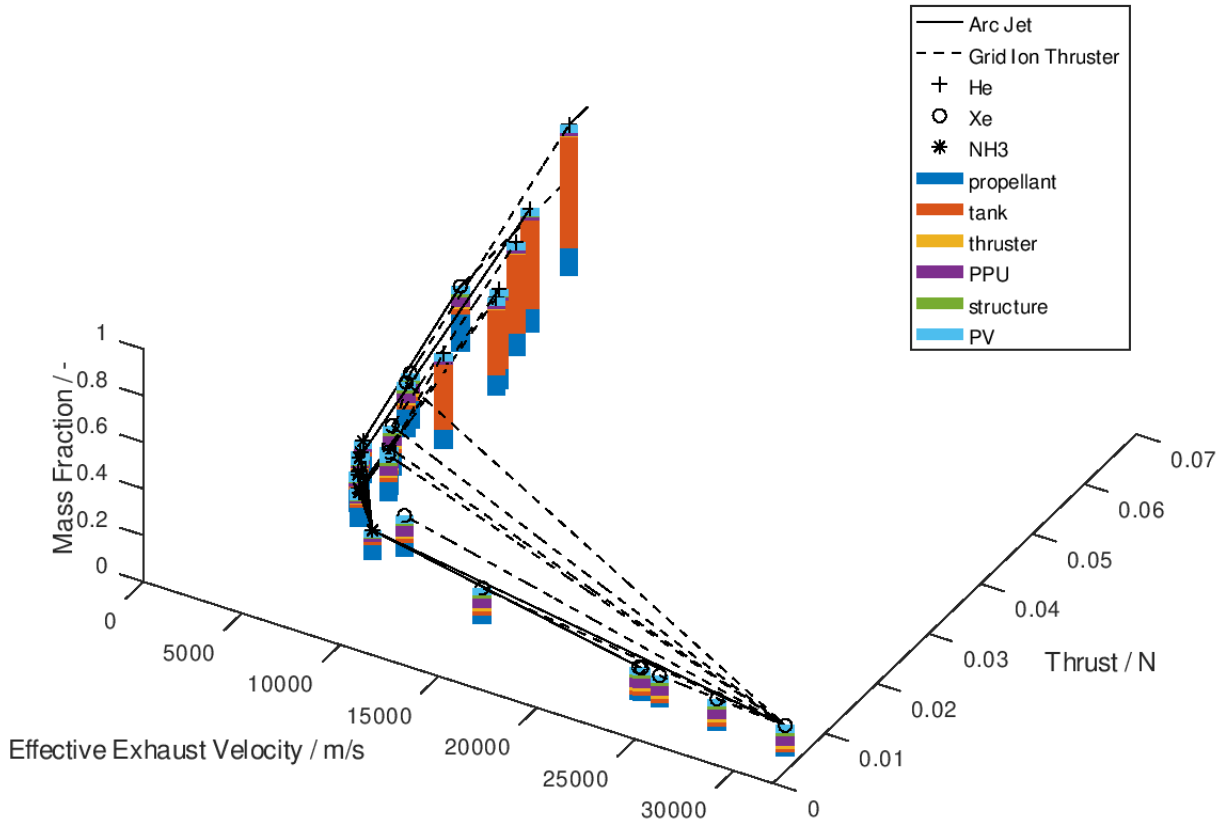

Figure 5. Evolutionary System Design Converger Example Output 3D mass fractions, in a c_e, F space including propulsion types and propellant indicators of the 18 best lineages for $\Delta v = 526$ m/s.

III. Example Application

Within the scope of the IRAS project a small satellite constellation is envisioned. A satellite total mass of $m_{total} = 150$ kg shall not be exceeded. Furthermore, a Δv of 526 m/s was deemed sufficient for the transfer from an insertion to an operational orbit[4,5]. The continuous power supplied to the electric propulsion system is limited to $P_{EP} = 202.9$ W. With these inputs, a simulation can be started for the ESDC. An example output is given in Fig. 5, where the three best resulting lineages/designs are displayed. Eleven dimensions of the design space are displayed, where seven dimensions are packed into the z-axis, as these are concerned with mass or mass fractions of all implemented subsystems. One bar makes up the total mass fraction, which the EP system would require of the total permitted mass. The relation of each individual bar size shows

the dominant part of each system. For example, helium based arcjets, marked by a "+" on top, require all a large tank mass indicated by large orange bars, as the required helium necessitates a large volume. On the other hand, ammonia based arcjets, marked by a "*", require fairly large propellant volumes, marked by their blue bars in comparison to their total mass.
Grid ion thrusters based on xenon are marked by a "o", which have a fairly even distribution among the systems, as propellant is utilized very efficiently.
Figure 5 shows clearly the evolutionary aspects of the algorithm of the ESDC. Successive changes in configuration and incremental steps allow to converge to an optimal solution.
Consequently, the XML data output yields the following data. An ammonia based arcjet thruster will require a total mass of approximately 19.5 kg, of which 10.1 kg are ammonia propellant, 2.0 kg are required as tank mass, 0.3 kg as thruster mass, 1.9 kg for the power processing unit, 0.5 kg for additional structure and 3.2 kg for respective solar panels. Thus, this system requires approximately 12.6 % of the total permitted mass of the satellite.
In a next step, the initial guess that produced the maximum spacecraft mass of 155 kg might be reduced to a new lower value and the ESDC can perform another iteration. Looking only at the 3D graph might lead to the erroneous assumption that an grid ion thruster with very large c_e would be the most beneficial system. This is obscured due to the fact that most of the evolution takes places around the high c_e mark of an ammonia arcjet. An alternative scenario with increased Δv comes to the conclusion that a xenon based ion thruster is optimal to perform the transfer task.
Another type of output aids to the understanding the ESDC and genetic algorithms in general. Allowing to explain the results of Fig. 5, a 2D bar graph, given exemplary in Fig. 6, is provided. Here the results for multiple lineages are displayed according to their number of generation. At generation zero, a large variation exists due initial random seeding, while over time this variation becomes minimal and before 25 generations are reached most lineages have converged to a single solution. The bar at generation five corresponds to the remaining optimal candidate of a xenon ion thruster, but multiple lineages do successfully find the ammonia arcjet solution, which is slightly more advantageous.
The graph of Fig. 6 further demonstrates that convergence time for evolutionary algorithms is challenging to predict. Nonetheless, the ESDC is capable to produce the desired result that all lineages converge onto a single best solution. Automatic animation of all these plots is implemented.

IV. Conclusion

The improvement of previous versions of the Evolutionary System Design Converger has been presented. The interface for mission specific input parameters has been simplified, additional inputs for adjusting the quality and performance of a simulation have been implemented as well as an extensive module to define desired output variants. All these inputs are to be provided with XML-files to offer a machine useable interface.
Scaling of performance and physical hardware is now fully generic and updates itself automatically from a provided database. This database has been further extended by publicly available data.
The software tool has been adapted to operate fully open-source, with Octave 4.4.1 and above. The ESDC is available on GitHub on terms of a MIT license equipped with a simple example database. Extending the database with proprietary data is a task for third-party users.
The new visualisation module of the ESDC allows for fast analysis of evolutionary data in n dimensions for multiple lineages during their full evolutionary history.
The ESDC is operating on a dedicated server as part of the Digital Concurrent Engineering Platform (DCEP) of the Integrated Research Platform of Affordable Satellite (IRAS), where it provides systems engineering solution currently focused on electric propulsion system data.

Acknowledgements

This work is funden by the state ministry of Baden-Württemberg for Wirtschaft, Arbeit und Wohnungsbau grant numbers 3-4332.62-DLR/49 (IRAS) and 3-4332.62-DLR/56 (IRAS II).
The work on the visualisation modules was fully supported by a scholarship through the Google Summer of Code 2019 program. This support is greatly appreciated.

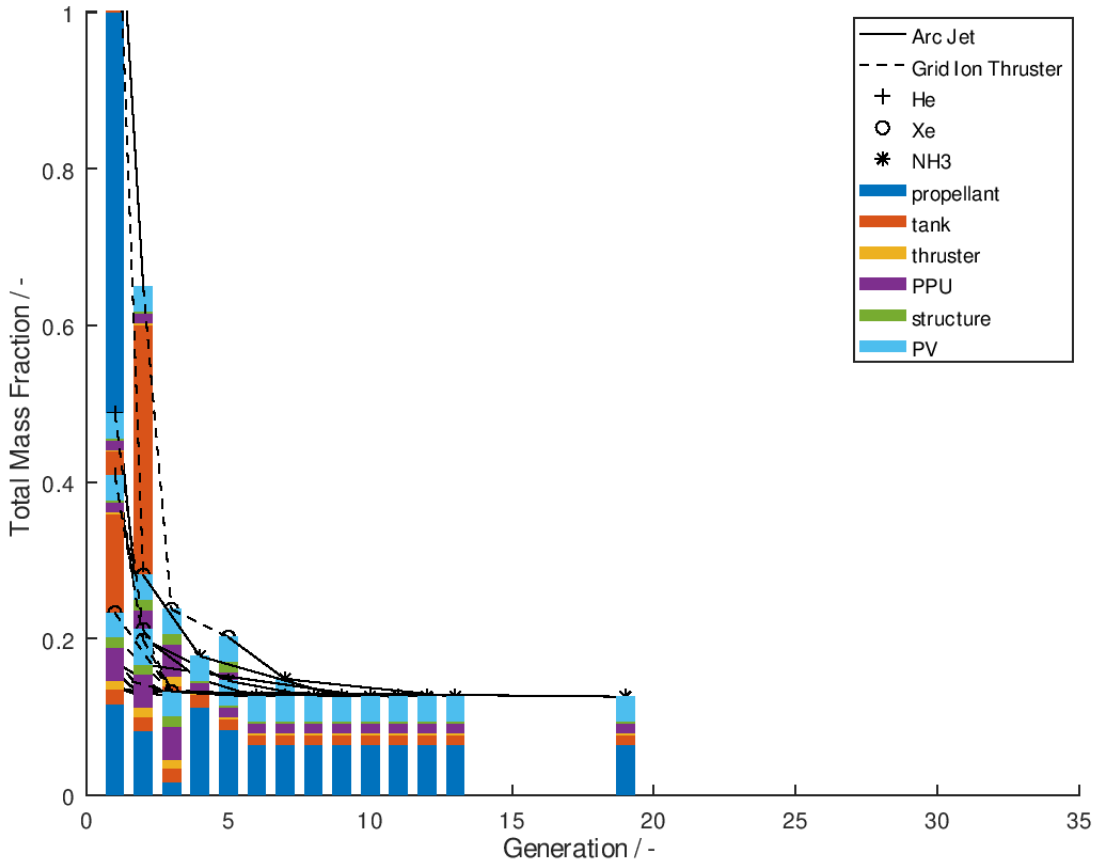

Figure 6. Evolutionary System Design Converger Example Output 2D mass fractions succession over generations for the 18 best lineages for $\Delta v = 526$ m/s.

References

[1]Ehresmann, M., and Spanoudis, T., "Evolutionary System Design Converger", GitHub Repository, https://github.com/aerospaceresearch/ESDC, cited 20 August 2019.

[2]Gilat, Amos, "MATLAB: An Introduction with Applications 2nd Edition", John Wiley & Sons, ISBN 978-0-471-69420-5, 2004

[3]Eaton, J.W.: GNU OCTAVE, https://octave.org/doc/interpreter/, cited 20 August 2019.

[4] Ehresmann, M., Skalden, J., Fugmann, M., Harmansa, N. , Herdrich, G., Fasoulas, S, Klinkner, S., Stäbler, T., Hümbert, S., Refle, O.: "IRAS: Low-cost Constellation Satellite Design, Electric Propulsion and Concurrent Engineering", 18,B4,6A,12,x44499, 69th International Astronautical Congress, Bremen, Germany, 2018

[5] Ehresmann, M., Skalden, J., Herdrich, G., Fasoulas, S.: "Automated System Analysis and Design for Electric Propulsion Systems, 344, Space Propulsion 2018, Seville, Spain, 2018

[5] Google: "Google Summer of Code", https://summerofcode.withgoogle.com/, cited 20 August 2019.