

Structural Dynam(o)ite

Optimized Design and Fabrication Workflows with Dynamo

Dieter Vermeulen, MSc. Str.Eng.
Technical Specialist AEC – Computational Design & Engineering
Autodesk

Learning Objectives

- Understand how to connect Dynamo Studio to Inventor Professional
- Learn how to use a skeleton-workflow approach to generate geometry
- Learn how to capitalize on the capabilities of the improved Frame Generator inside Inventor
- Learn how to export from Inventor to Revit to close the gap between manufacturing and architecture

Description

In the world of structural engineering, we are challenged to consider many design options, to find the fittest solution in the shortest time, and to be leaders in economic structure design. In this class, you will discover how computational design techniques can help automate this process in design and fabrication workflows for building structures. You will be blown away by several structural engineering workflows that you can use to optimize your designs with Dynamo. All kinds of techniques and computational design approaches will be considered to generate, evaluate, evolve, and cycle through structural design processes. You will understand how to perform your own design automation for repetitive structural tasks.

Speaker

Dieter Vermeulen



Working as a Technical Specialist AEC for the Northern European region at Autodesk, I'm specialized in the products of the Computational Design and Engineering portfolio. Within that domain I support the authorized Autodesk channel partners and our customers with innovative workflows and solution strategies. I evangelize the power of computational design with Dynamo in the building and infrastructure industry. This results in workflows covering the process from design, analysis, construction to fabrication for structural steel and reinforced concrete structures in building and infrastructure projects.



Twitter

www.twitter.com/BIM4Struc



LinkedIn

www.linkedin.com/in/dietervermeulen



WordPress personal blog

www.revitbeyondbim.wordpress.com



Typepad team blog

www.autodesk.typepad.com/bimtoolbox/



YouTube Channel

www.youtube.com/user/RevitbeyondBIM



AU Online Profile

<http://au.autodesk.com/au-online/profile?code=Ly7Q7iB6RnpX5p/oiyTtRw%3D%3D>

Table of Contents

Learning Objectives	1
Description	1
Speaker	2
Table of Contents.....	3
Disclaimer	4
Computational Design.....	5
Connected Outcome	5
Autodesk Dynamo.....	5
Optimization	6
Multi-Objective Optimization	6
Optimization Techniques	8
Software Requirements.....	12
Dynamo Packages	13
Project Refinery	14
Project Fractal	15
Spatial Truss Deformation.....	16
Project Description.....	16
Geometry Definition	17
Deformation analysis	20
Results evaluation.....	23
Optimization in Refinery	24
Visualization of Geometry.....	27
Truss Shape Optimization	28
Project Description.....	28
Method 1 – Design Exploration	30
Method 2 – Brute Force Search	38
Genetic Algorithm Optimization	46
Mars Torus Arena	53
Computational Model	53
Structure Evaluation	54
Datasets.....	55
Other references	56

Disclaimer

This presentation may make statements regarding future events and development efforts for our products and services.

These statements reflect our current expectations based on what we know today.

Our plans are not intended to be a promise or guarantee of future delivery of products, services or features and purchasing decisions should not be made based upon these statements.

We do not assume any responsibility to update this roadmap to reflect events that occur or circumstances that exist after the publish date of this roadmap.

Computational Design

Connected Outcomes

Outcome based BIM technology helps generate a virtually infinite number of design alternatives. Start with your desired outcome, then explore a near infinite range of possibilities and produce optimal designs in a fraction of the time - the power of the cloud helps make this a reality today. Designers and engineers can generate multiple options, ideas and scenarios more rapidly than ever before - exploring forms, simulating performance, and impressing clients with optimal designs.

Computational Design for BIM is an intelligent model-based process that provides a framework for negotiating and influencing the interrelation of internal and external building parameters.

In relation to design, computation involves the processing of information and interactions between elements which constitute an environment. Computational Design provides a framework for negotiating and influencing the interrelation of both internal and external properties, with the capacity to generate complex order, form, and structure. By combining the principles of computational design with Building Information Modeling, a fundamentally new method of building design is made possible.

Literally, “computational design” means “giving shape with the means of the power of a computer and scripting”.

Autodesk Dynamo

Autodesk gives an answer to this new challenge in our design world. This solution is called **Autodesk® Dynamo**. Dynamo lets designers and engineers create visual logic to explore parametric designs and automate tasks. It helps you to solve challenges faster by designing workflows that drive the geometry and behavior of design models. With Dynamo you will extend your designs into interoperable workflows for documentation, fabrication, coordination, simulation, and analysis.

More information can be found on www.dynamobim.org.

Optimization

Optimization is the selection of a best element (with regards to some criterion) from some set of available alternatives.

In its simplest form, an optimization problem consists of maximizing or minimizing a function by choosing input values from a given range and computing the value of the output function.

From above optimization definition we can conclude that optimization is about finding the best solution within a set of alternatives. Usually when thinking of “optimization” we have one objective in mind which is maximized or minimized. This is called Single Objective Optimization (SOO) and is the process of finding the single best solution to a problem. With SOO you want to find the best solution by either not taking any other factors of the problem into account, or by combining all the factors of the problem into one objective.

Optimization in the AEC industry rarely is a single objective problem. Whether working with design, engineering or construction, finding one optimal solution is rarely possible or desired. Usually, optimization involves multiple competing objectives, therefore optimization becomes a matter of finding the best trade-off between these objectives rather than finding the one best solution

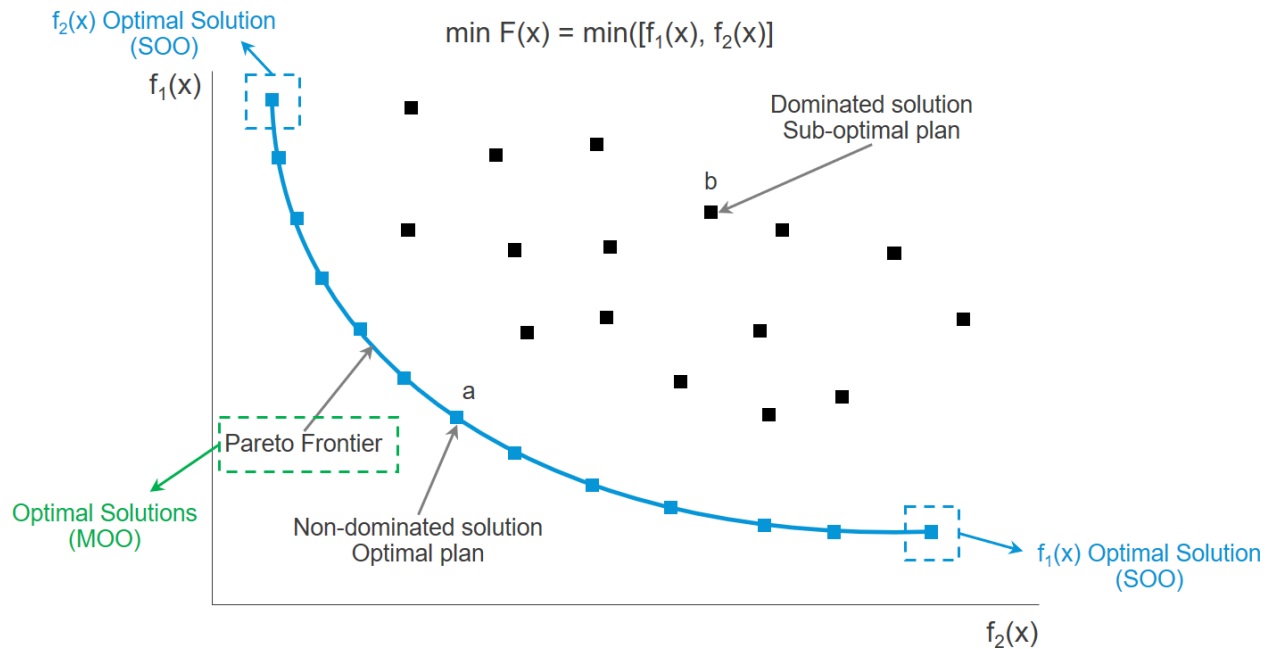
Multi-Objective Optimization

Multi-objective optimization (MOO) is an area of multiple criteria decision making, that is concerned with mathematical optimization problems involving more than one objective function to be optimized simultaneously. Multi-objective optimization is applied in many fields where optimal decisions need to be taken in the presence of trade-offs between two or more conflicting objectives.

Minimizing cost while maximizing comfort while buying a car and maximizing performance whilst minimizing fuel consumption and emission of pollutants of a vehicle are examples of multi-objective optimization problems involving two and three objectives, respectively. In practical problems, there can be more than three objectives.

For a nontrivial multi-objective optimization problem, no single solution exists that simultaneously optimizes each objective. In that case, the objective functions are said to be conflicting, and there exists a (possibly infinite) number of **Pareto optimal solutions**. A solution is called non-dominated or Pareto optimal if none of the objective functions can be improved in value without degrading some of the other objective values. Without additional subjective preference information, all Pareto optimal solutions are considered equally good (as vectors cannot be ordered completely). The goal may be to find a representative set of Pareto optimal solutions, and/or quantify the trade-offs in satisfying the different objectives, and/or finding a single solution that satisfies the subjective preferences of a human decision maker.

For a design to be in the Pareto optimal set it cannot be dominated by another solution. If a solution is worse than another solution on all objectives, then it is dominated and not in the Pareto optimal set.

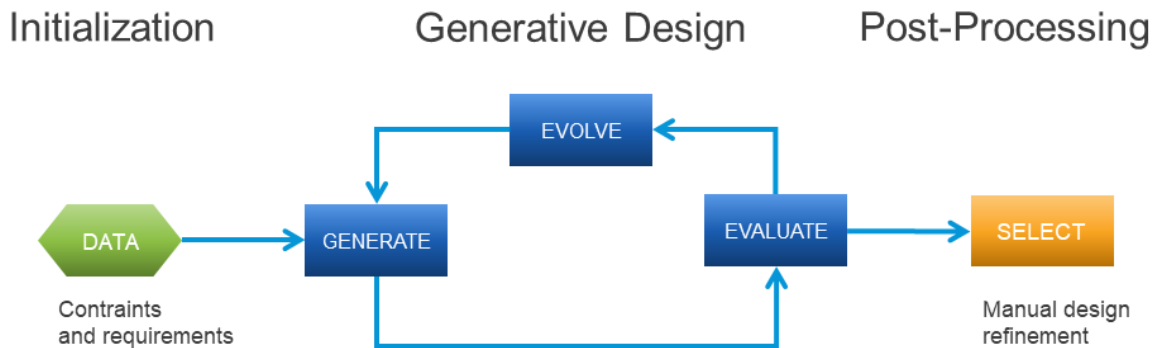


The curve connecting all non-dominated solutions (see figure above) is known as the **pareto frontier**. Deciding which solution on the pareto frontier is the favorite design, is up to the decision maker. When dealing with MOO problems some information is usually missing. Desirable objectives are given, but the final information that separates two solutions on the pareto frontier should be weighted by the decision maker. Often, the missing information is something that is difficult to measure like constructability and aesthetics, and therefore the decision-making needs human interaction.

Optimization Techniques

Generative Design

Generative design is an iterative design process that involves a program that will generate a certain number of outputs that meet certain constraints, and a designer that will fine tune the feasible region by changing minimal and maximal values of an interval in which a variable of the program meets the set of constraints, in order to reduce or augment the number of outputs to choose from.



Often, generative design used **Genetic Algorithms (GA)** to perform the “evolve” part.

Genetic Algorithms

A Genetic Algorithm (GA) is a search-based optimization technique based on the principles of natural selection. GA's are often used to find optimal or near optimal solutions to complicated projects, where a brute force method would take too long and use too much computing power.

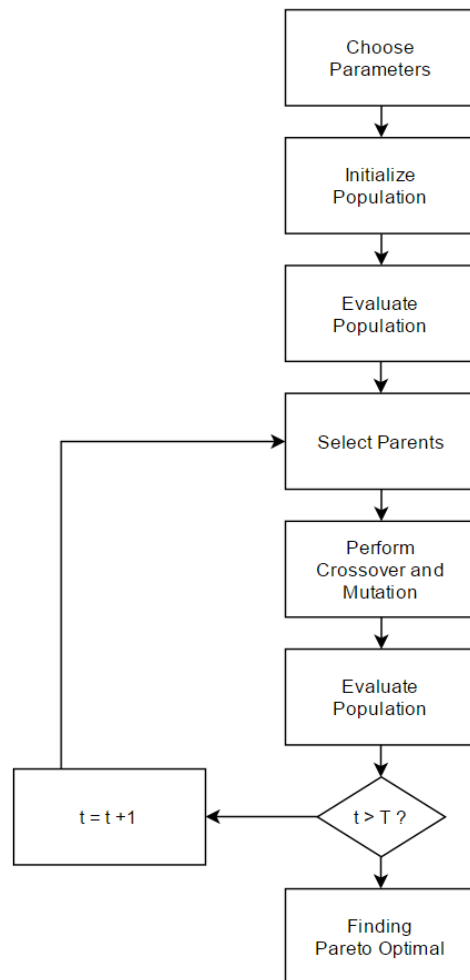
A GA is a great method to solve MOO problems with, because you can localize more pareto optimal solutions in a large solution space, in an efficient way. Let's look at a simple example of why a search-based optimization technique, like a GA, can be ideal to use for solving MOO problems.

“I ’m thinking of a number between one and one billion. How long will it take for you to guess it? Solving a problem with “brute force” refers to the process of checking every possible solution. Is it one? Is it two? Is it three? Is it four? And so forth. Though luck does play a factor here, with brute force we would often find ourselves patiently waiting for years while you count to one billion. However, what if I could tell you if an answer you gave was good or bad? Warm or cold? Very warm? Hot? Super, super cold? If you could evaluate how “fit ” a guess is , you could pick other numbers closer to that guess and arrive at the answer more quickly. Your answer could evolve.

Source: <https://natureofcode.com/book/chapter-9-the-evolution-of-code/>

The tools used for this class, make use of the NSGA-II (Non-dominated Sorting Genetic Algorithm-II) evolutionary algorithm.

A typical workflow in a structural optimization process is represented in the flowchart below:



Initialization

The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. Often, the initial population is generated randomly, allowing the entire range of possible solutions (the search space). Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.

Selection

During each successive generation, a portion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select the best solutions. Other methods rate only a random sample of the population, as the former process may be very time-consuming.

The fitness function is defined over the genetic representation and measures the quality of the represented solution. The fitness function is always problem dependent.

For instance, in the knapsack problem one wants to maximize the total value of objects that can be put in a knapsack of some fixed capacity. A representation of a solution might be an array of bits, where each bit represents a different object, and the value of the bit (0 or 1) represents whether the object is in the knapsack. Not every such representation is valid, as the size of objects may exceed the capacity of the knapsack. The fitness of the solution is the sum of values of all objects in the knapsack if the representation is valid, or 0 otherwise.

In some problems, it is hard or even impossible to define the fitness expression; in these cases, a simulation may be used to determine the fitness function value of a phenotype (e.g. computational fluid dynamics is used to determine the air resistance of a vehicle whose shape is encoded as the phenotype), or even interactive genetic algorithms are used.

Genetic Operators

The next step is to generate a second-generation population of solutions from those selected through a combination of genetic operators: crossover (also called recombination), and mutation.

For each new solution to be produced, a pair of "parent" solutions is selected for breeding from the pool selected previously. By producing a "child" solution using the above methods of crossover and mutation, a new solution is created which typically shares many of the characteristics of its "parents". New parents are selected for each new child, and the process continues until a new population of solutions of appropriate size is generated. Although reproduction methods that are based on the use of two parents are more "biology inspired", some research suggests that more than two "parents" generate higher quality chromosomes.

These processes ultimately result in the next generation population of chromosomes that is different from the initial generation. Generally, the average fitness will have increased by this procedure for the population, since only the best organisms from the first generation are selected for breeding, along with a small proportion of less fit solutions. These less fit solutions ensure genetic diversity within the genetic pool of the parents and therefore ensure the genetic diversity of the subsequent generation of children.

Heuristics

In addition to the main operators above, other heuristics may be employed to make the calculation faster or more robust. The speciation heuristic penalizes crossover between candidate solutions that are too similar; this encourages population diversity and helps prevent premature convergence to a less optimal solution.

Termination

This generational process is repeated until a termination condition has been reached. Common terminating conditions are:

- A solution is found that satisfies minimum criteria
- Fixed number of generations reached
- The highest-ranking solution's fitness is reaching or has reached a plateau such that successive iterations no longer produce better results

Brute-force search

In computer science, brute-force search or exhaustive search, also known as generate and test, is a very general problem-solving technique and algorithmic paradigm that consists of systematically enumerating all possible candidates for the solution and checking whether each candidate satisfies the problem's statement.

While a brute-force search is simple to implement, and will always find a solution if it exists, its cost is proportional to the number of candidate solutions – which in many practical problems tends to grow very quickly as the size of the problem increases (combinatorial explosion).

Therefore, brute-force search is typically used when the problem size is limited, or when there are problem-specific heuristics that can be used to reduce the set of candidate solutions to a manageable size. The method is also used when the simplicity of implementation is more important than speed.

Software Requirements

Before playing around with the delivered datasets, make sure the following software is available on your computer.

- **Autodesk Robot™ Structural Analysis Professional 2019**

Used for the structural analysis of the trusses

More info on: <http://www.autodesk.com/products/robot-structural-analysis/overview>

- **Autodesk Dynamo 2.1.0.6544**

Free open-source tool for the computational design and optimization of the structure

More info on: <http://www.dynamobim.org>

- **Project Refinery 0.3.53**

Project Refinery allows users to perform design exploration including optioneering and optimization workflows on Dynamo projects.

Read more [here](#).

- **Project Fractal**

This online service allows a user to explore the parametric design space of models created in Dynamo Studio with the automatic generation of a wide sampling of options.

Read more [here](#).

Dynamo Packages

Make sure the following packages are installed in Dynamo:

- **Optimo for Dynamo 0.1.2**
Package (plugin) for Dynamo with Genetic Algorithm nodes for the optimization of design and analysis.
More info on: https://github.com/BPOpt/Optimo/wiki/0_-Home
- **Structural Analysis for Dynamo 0.2.2**
Package which makes the connection between Dynamo and Robot Structural Analysis. More information about the Structural Analysis for Dynamo package can be found in the handout of [this class](#) at Autodesk University 2015 (ES9542 – Dynam(o)ite Your Design for Engineers)
- **BIM4Struc.Productivity 2.1.3**
Package that contains additional functionalities for structural design and evaluation.
- **DynaShape 0.5.1.1**
Dynamo package for constraint-based form finding, optimization and physics simulation. Installation instructions: <https://forum.dynamobim.com/t/dynashape/11666>

Learn more about the package from this AU Class:

<http://au.autodesk.com/au-online/classes-on-demand/class-catalog/classes/year-2018/dynamo-studio/aulon32162#chapter=0>

For the use of DynaShape with Refinery, you will need the special DynaShape build where all the live interactive feature has been stripped away so that the package can run in CLI mode.

1. Download the custom package via [this link](#).

2. Open the folder `C:\Users\[YOUR USERNAME]\AppData\Roaming\Dynamo\Dynamo Core\[YOUR DYNAMO VERSION]\packages`

3. Replace all the content of the subfolder `_DynaShape` with the content from this new downloaded package.

Optionally do the same with

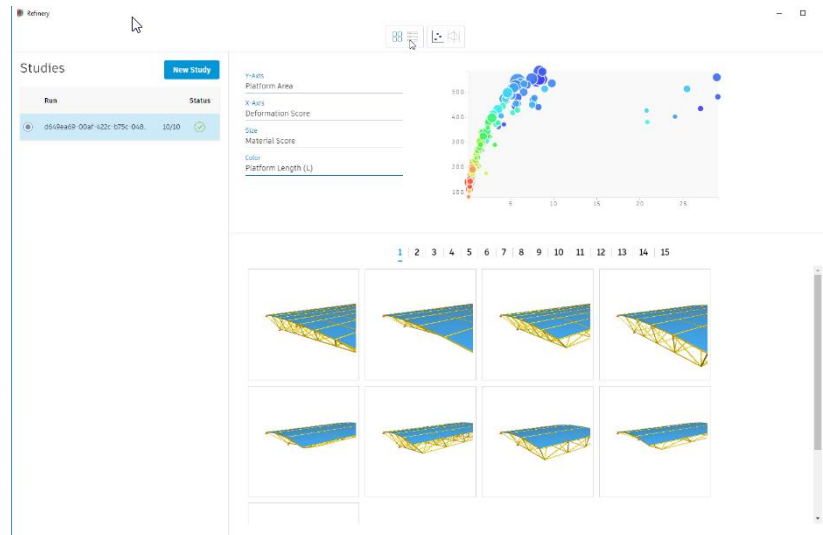
`C:\Users\[YOUR USERNAME]\AppData\Roaming\Dynamo\Dynamo Revit\[YOUR DYNAMO VERSION]\packages`

- **Archi-lab Mandrill 2018.2.2**
Package for data visualization in Dynamo.
More information: <http://archi-lab.net/category/dynamo/mandrill/>

Project Refinery

Project Refinery allows users to perform design exploration including optioneering and optimization workflows on Dynamo projects.

Refinery uses the NSGA-II optimization algorithm (a type of genetic algorithm), which is a meta-heuristic optimization algorithm for multi-objective optimization. Genetic algorithms are commonly used to generate high-quality solutions to optimization and search problems by relying on bio-inspired operators such as mutation, crossover and selection. A genetic algorithm is a population-based optimization. Population-based approaches maintain and improve multiple candidate solutions, often using population characteristics to guide the search. Each round of the optimization processed in the genetic algorithm is called a generation. So, when you set the generation value to 40, the process of selection, cross over, and mutation happens 40 times for each population of 40 designs. You can read more here. The seed simply says where the algorithm should start. If you keep the seed at "64" and then do another run with the same inputs you should get the same answer.

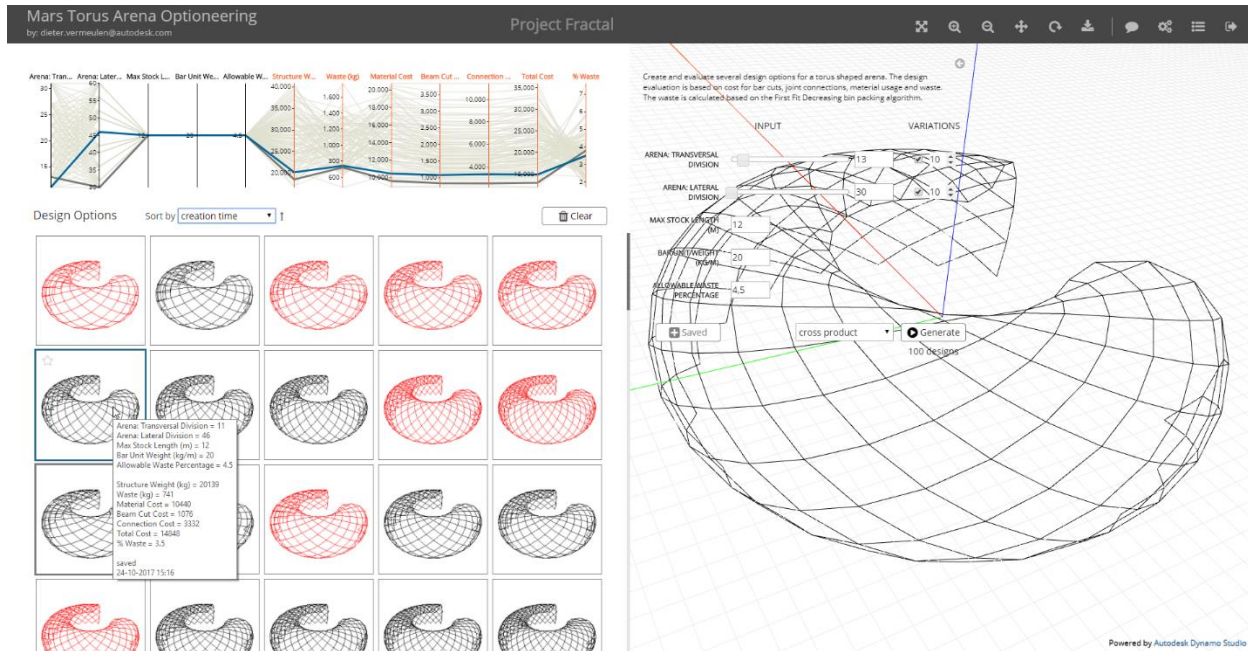


Important notes about Project Refinery:

- Connects with Dynamo 2.0
- Design Exploration software
- Optioneering and Optimization workflows
- Advanced results display
- Syncing selected option back to Dynamo
- Request access through this link: <https://beta.autodesk.com/key/refinery>

Project Fractal

Generating options in a “brute-force” method can be done with the beta project “Project Fractal”. This online service allows a user to explore the parametric design space of models created in Dynamo Studio with the automatic generation of a wide sampling of options.



Important notes about Project Fractal:

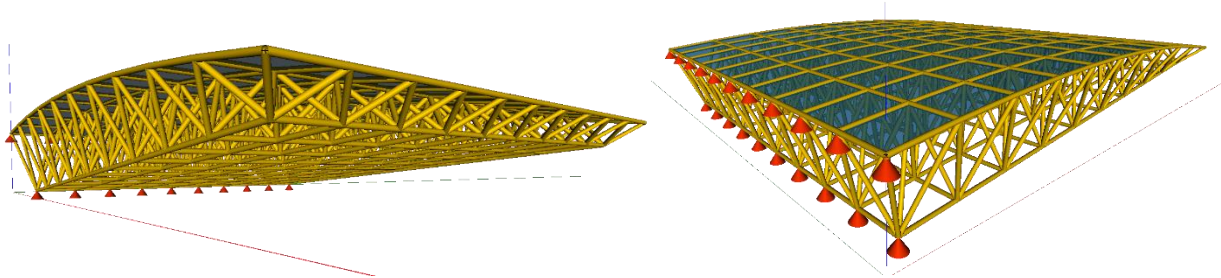
- It is not a commercial service or product
- Only for testing purposes / feedback gathering
- Connects with Dynamo Studio
- Design Exploration software that supports optioneering workflows

Request access through this link: <https://home.fractal.live/>

Spatial Truss Deformation

Project Description

In this case study a conceptual analysis is performed for the deformation of a spatial truss structure for a cantilever roof.

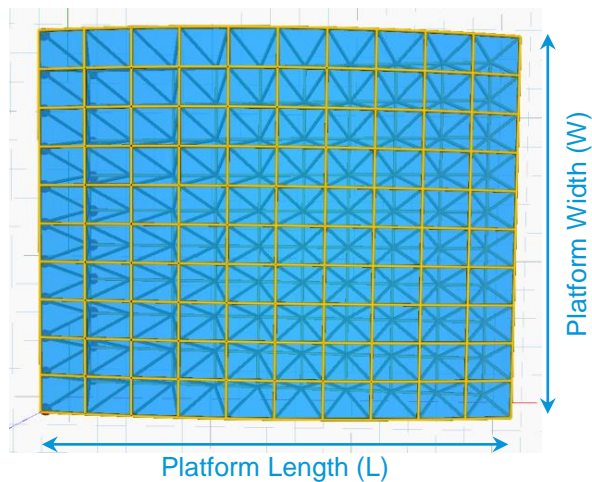
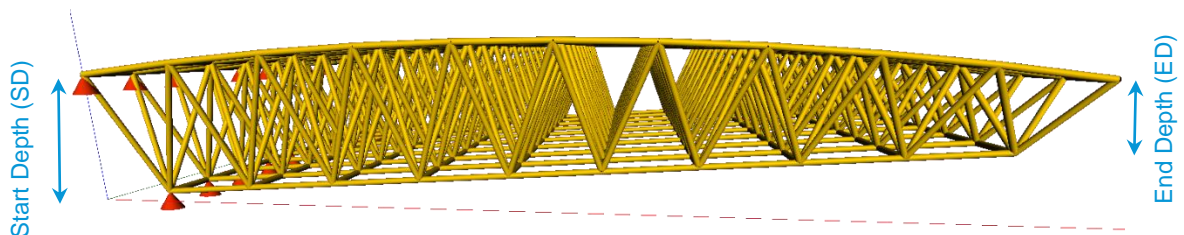


The optimization problem consists of finding the best truss configuration with three objectives:

- Maximize Platform Area
- Minimize Deformation
- Minimize Structure Weight

The configuration of the truss can be changed, by varying these inputs:

- Height of the truss at start
- Height of the truss at end
- Truss Divisions = number of panels/squares along the width

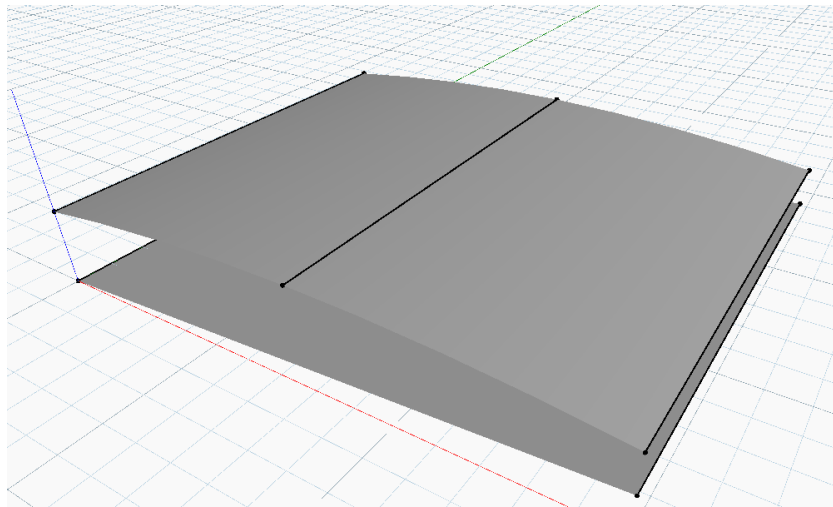
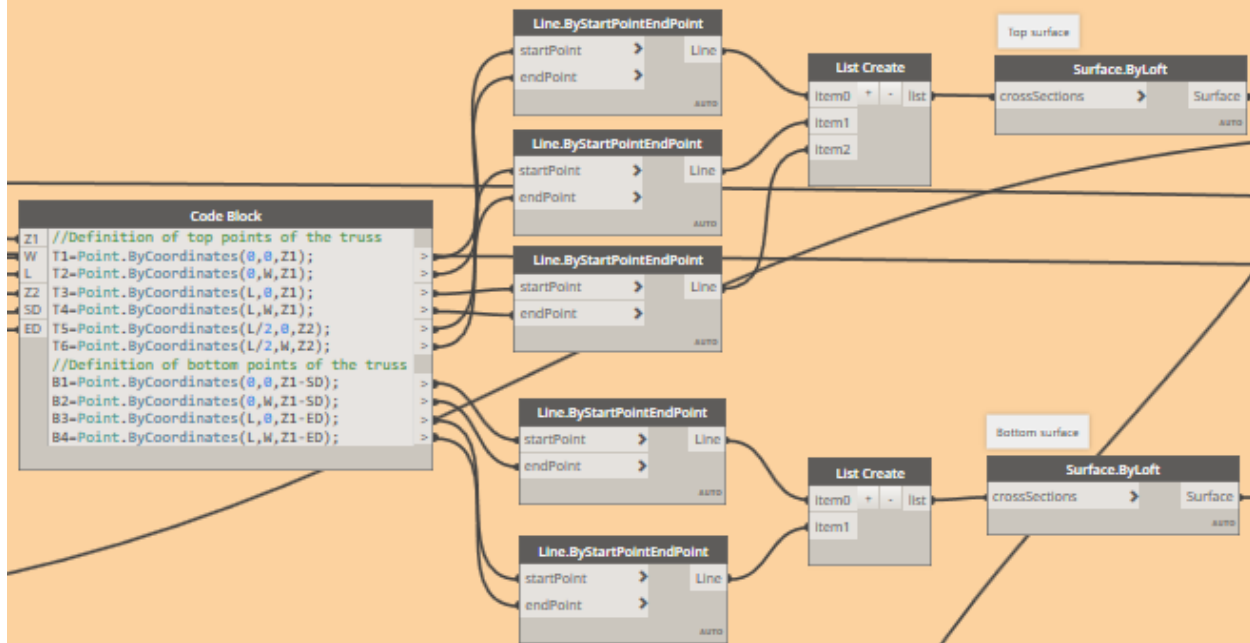


Geometry Definition

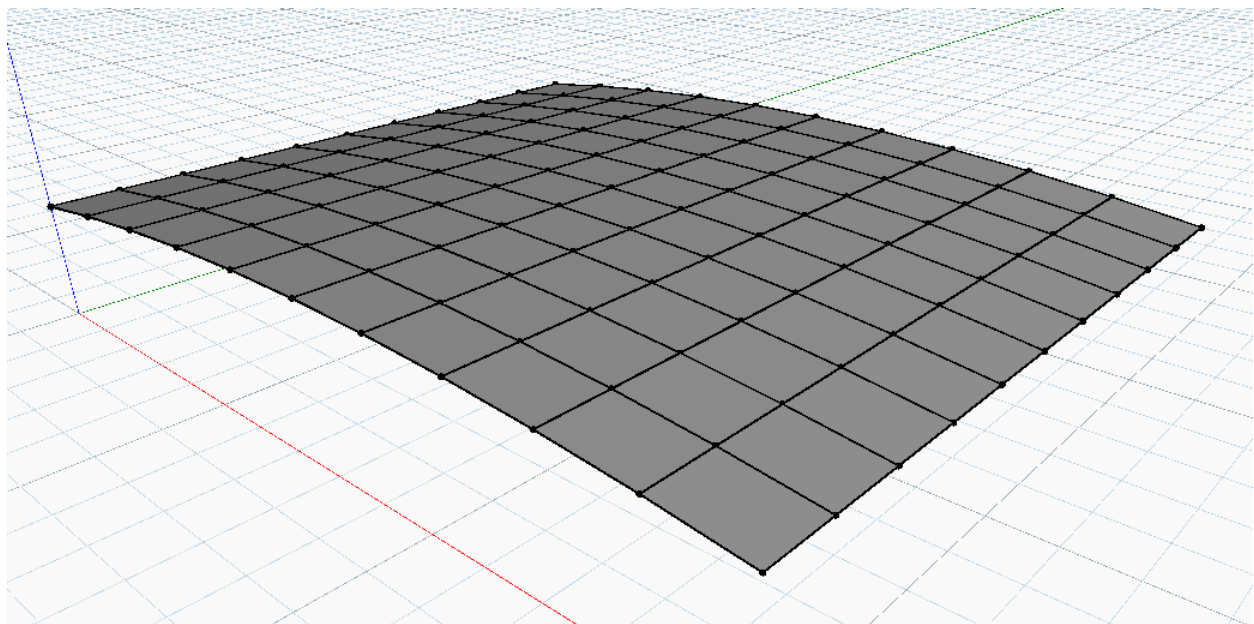
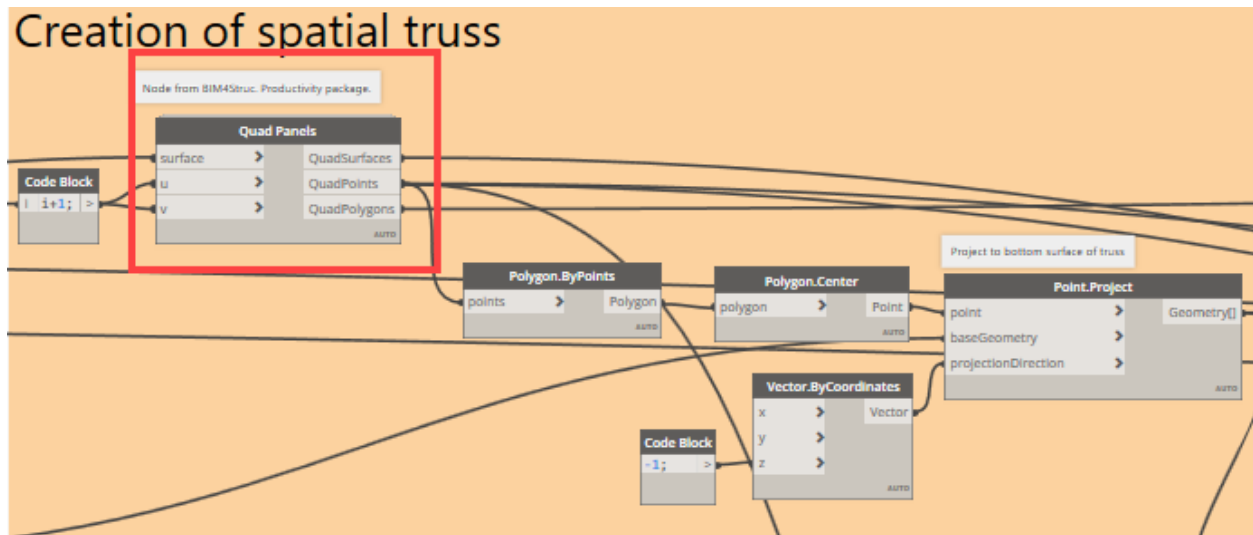
The top roof surface is defined by generating lines and loft them to a curved roof surface. The lines are defined by points, plotted with specific coordinates depending on the platform length (L), platform width (W), truss start depth (SD), truss end depth, and the platform top levels (Z1 and Z2).

The bottom surface is defined in a similar way and will be used to project the bottom of the truss division.

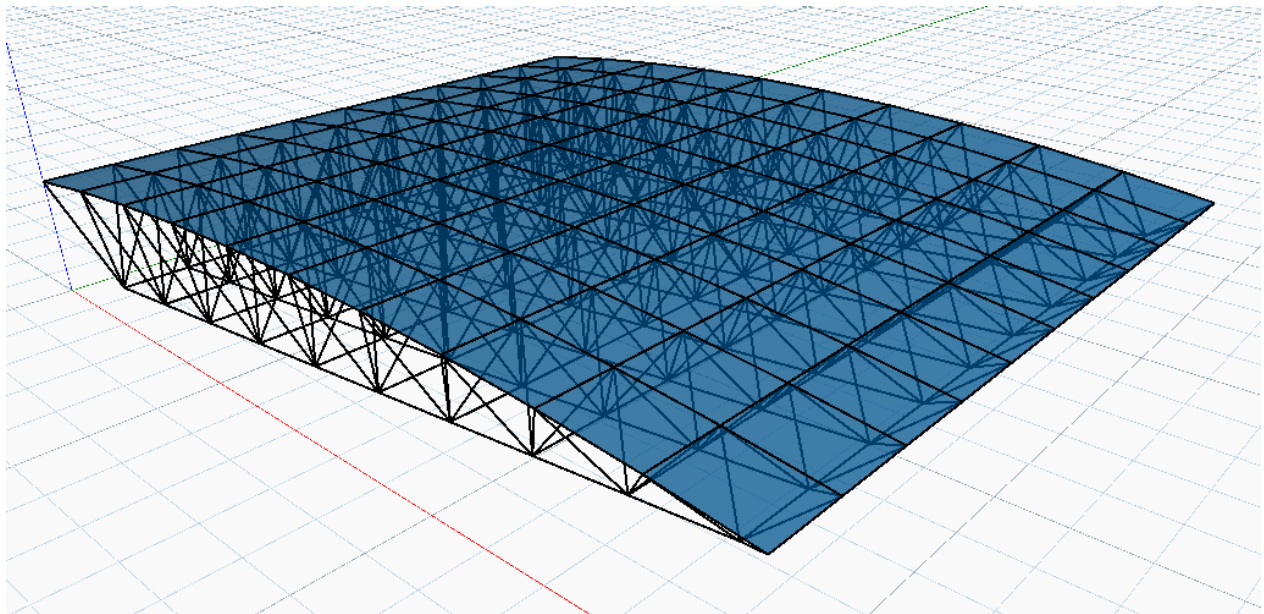
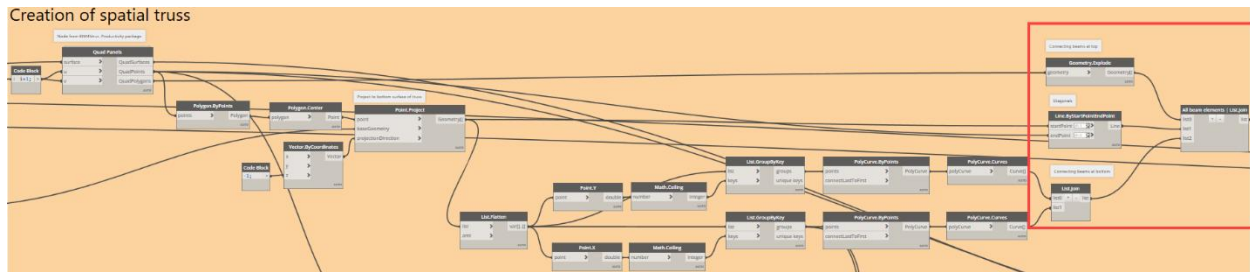
Definition of geometrical boundaries



With the custom node *Quad Panels* from the *BIM4Struc.Productivity* package, quad panels are created from the top surface.

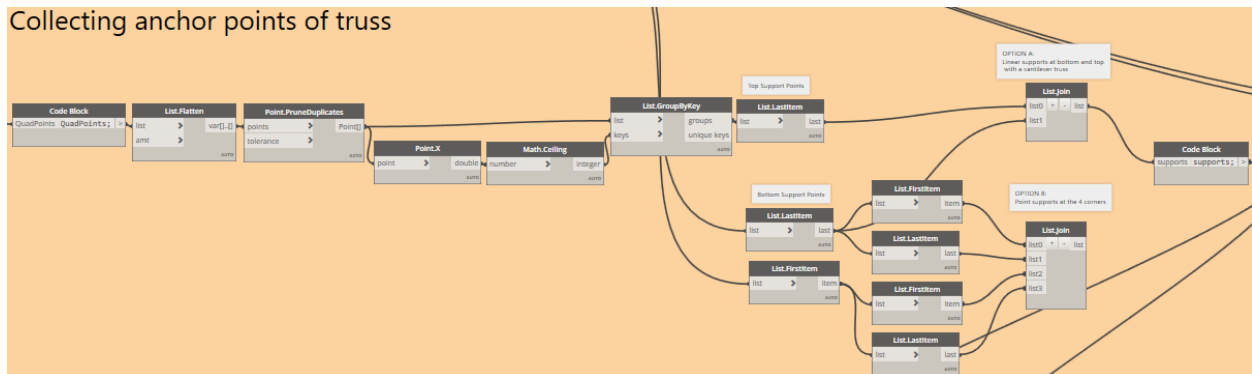


The points from the quad panels are projected to the bottom surface and then reused to create the spatial truss lines, by using sorting methods, based on the X and Y axis positions of the points at top and bottom.



The conceptual analysis of the deformation is performed with the DynaShape package. This package allows the user to perform constraint-based form finding, optimization and physics simulation.

Collecting anchor points of truss



DynaShape goal defining the supporting points to NOT displace. The value of "weight" defines how strong the anchor positions are. The greater the value, the less movement possible.

Goals.AnchorGoal_Create

startPosition	>	var
anchor	>	var
weight	>	var

Code Block: 1000000; >

The *AnchorGoal* simulates a support by keeping the truss node at a specified anchor point. By default, the weight for this goal is set very high to ensure the node really “sticks” to the anchor.

DynaShape goal defining that the beam elements can not be stretched.
By increasing the "weight" value, the material stiffness increases.

Goals.LengthGoal_Create

line	>	var
targetLength	>	
weight	>	

AUTO

Curve.Length

curve	>	double
-------	---	--------

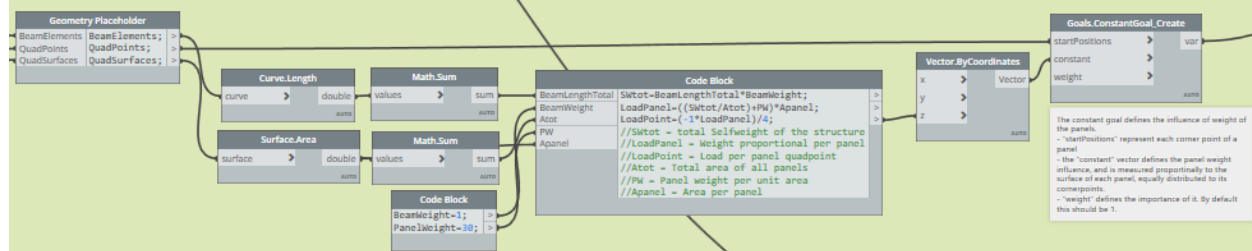
AUTO

Code Block

```
50000; >
```

The *LengthGoal* makes it possible to maintain the specified distance between two nodes located at the start and end point of the given line. When you increase the value of *weight* you simulate an increase in stiffness of the line.

Loads on top elements

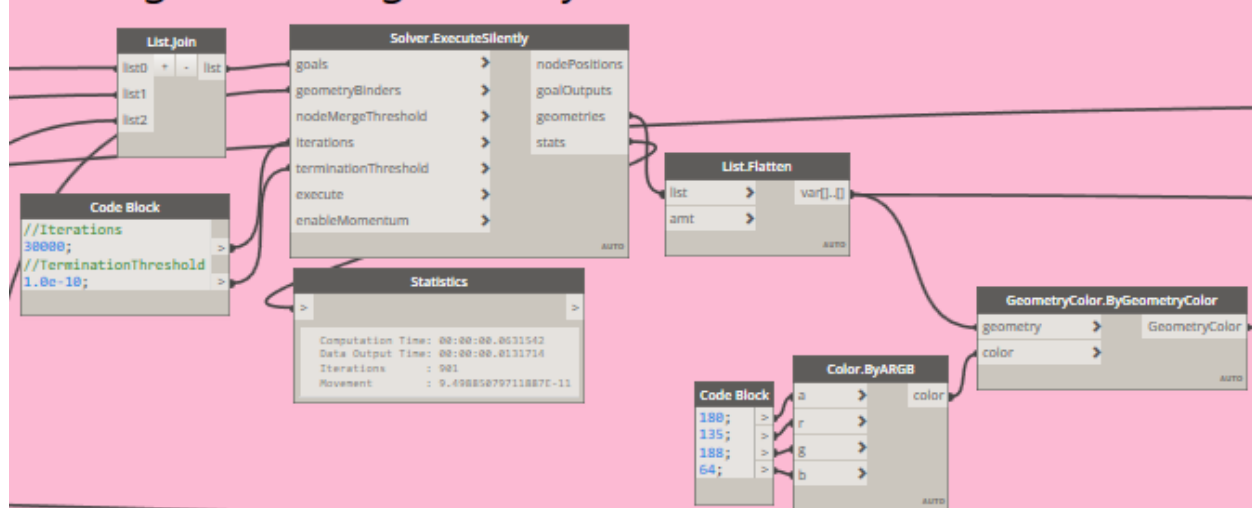


The *ConstantGoal* applies a constant directional offset to the specified points. For example, this is useful to simulate gravity.

The loads here are defined by the selfweight of a beam and the weight of a panel. This weight is distributed to the four corner points of a quad panel.

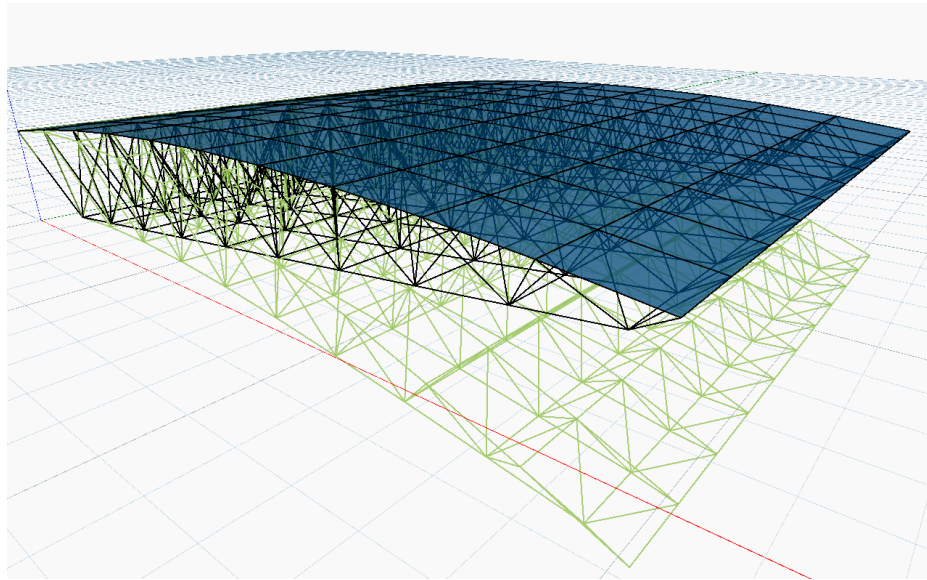
The weight will change depending on the platform dimensions.

Solving the truss geometry behavior



The solver executes in silent mode. This is needed to make it compatible with Refinery in a later phase.

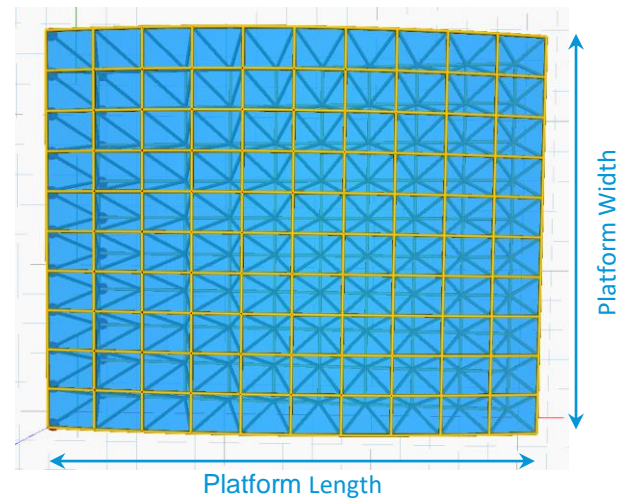
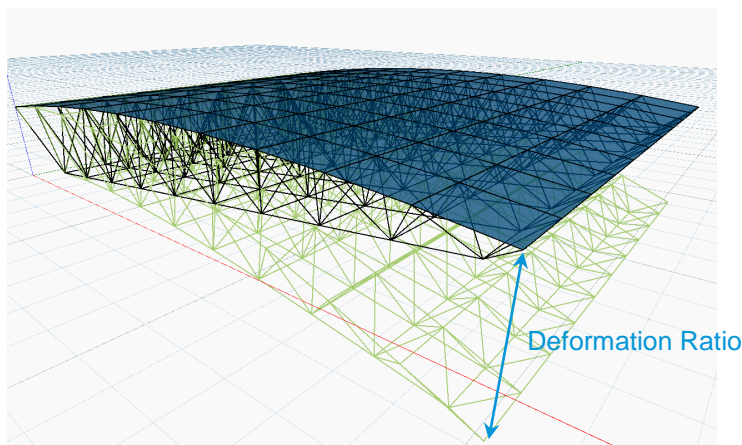
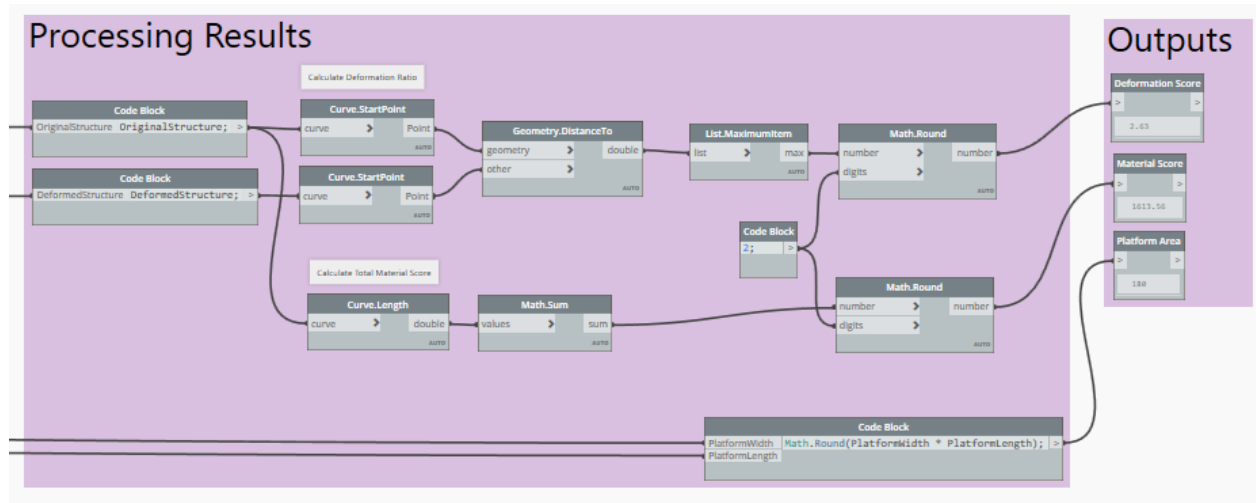
The *GeometryColor* node changes the color of the resulting geometry, to see a difference between the original structure (black) and the deformed shape (green). This deformation is influenced by the goals defined in Step 2 above.



Results evaluation

The results from this analysis are used for minimizing the objectives.

- **Deformation Ratio:**
Difference between original structure and structure after DynaShape analysis
- **Material Score:**
Total material used, by making the sum of all curve lengths
- **Platform Area:**
 $\text{Platform Area} = \text{Platform Width} \times \text{Platform Length}$



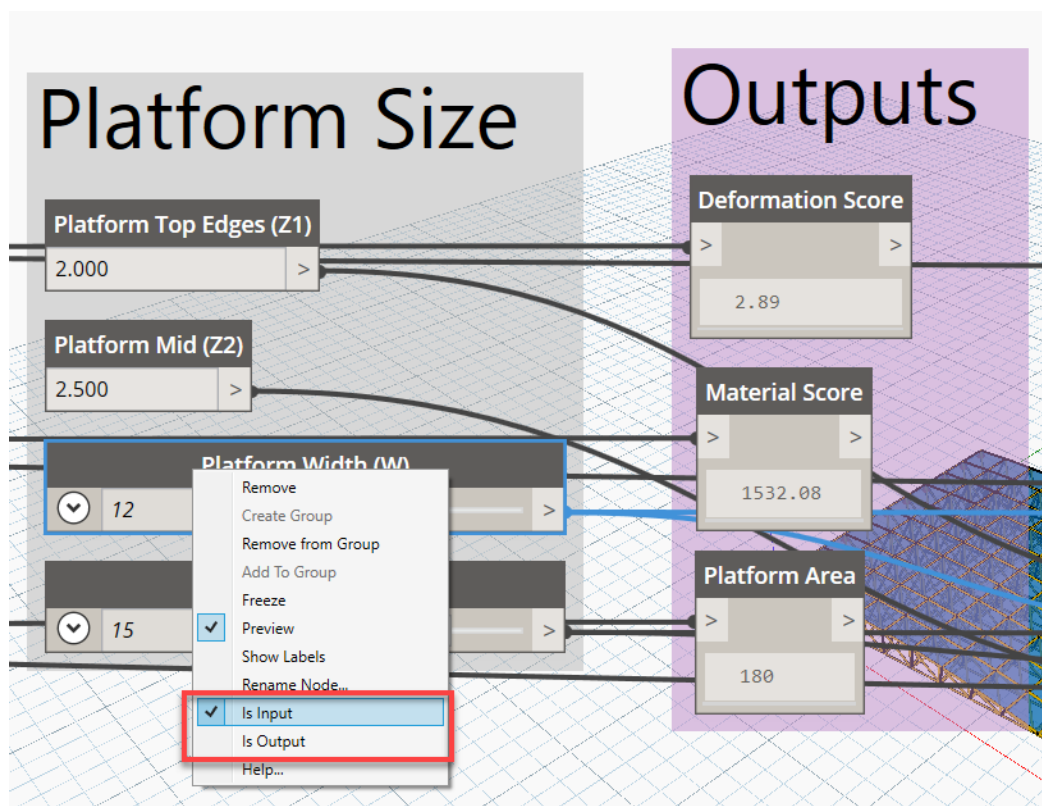
Optimization in Refinery

The script can be run in a randomization (for brute-force search) or optimization process with *Project Refinery*.

The inputs and outputs needed for Refinery need to be defined in the Dynamo script.

The inputs can be sliders. They need to be indicated as “Is Input” in the contextual menu of the node (right-mouse click on the node).

The outputs need to be named *Watch* nodes. Naming the nodes can be done by double-clicking the header and change the name. Besides that, the node needs to be set as “Is Output” in the contextual menu of the node.



Before you create a new study in Refinery, make sure the script has been executed manually and the script is saved.

1. Open Refinery in Dynamo in the menu View > Refinery...
2. Select the *Generation Method*
3. Expand *Inputs* and choose which inputs participate in automated runs
4. Expand *Outputs* and choose what optimization criteria should be applied.
5. Expand *Settings* to define the generation criteria.
6. Click *Generate* to start the run.

Create Study ×

Generation Method
Optimize

Generate

Inputs

Truss Start Depth (SD)

2

0.1

3

Truss End Depth (ED)

0.5

0.1

3

Truss Segments (S)

10

3

15

Platform Width (W)

12

8

30

Platform Length (L)

15

5

20

Outputs

Create Study ×

Generation Method
Optimize

Generate

Inputs

Outputs

Material Score

1532.08

MINIMIZE

Deformation Sco...

2.89

MINIMIZE

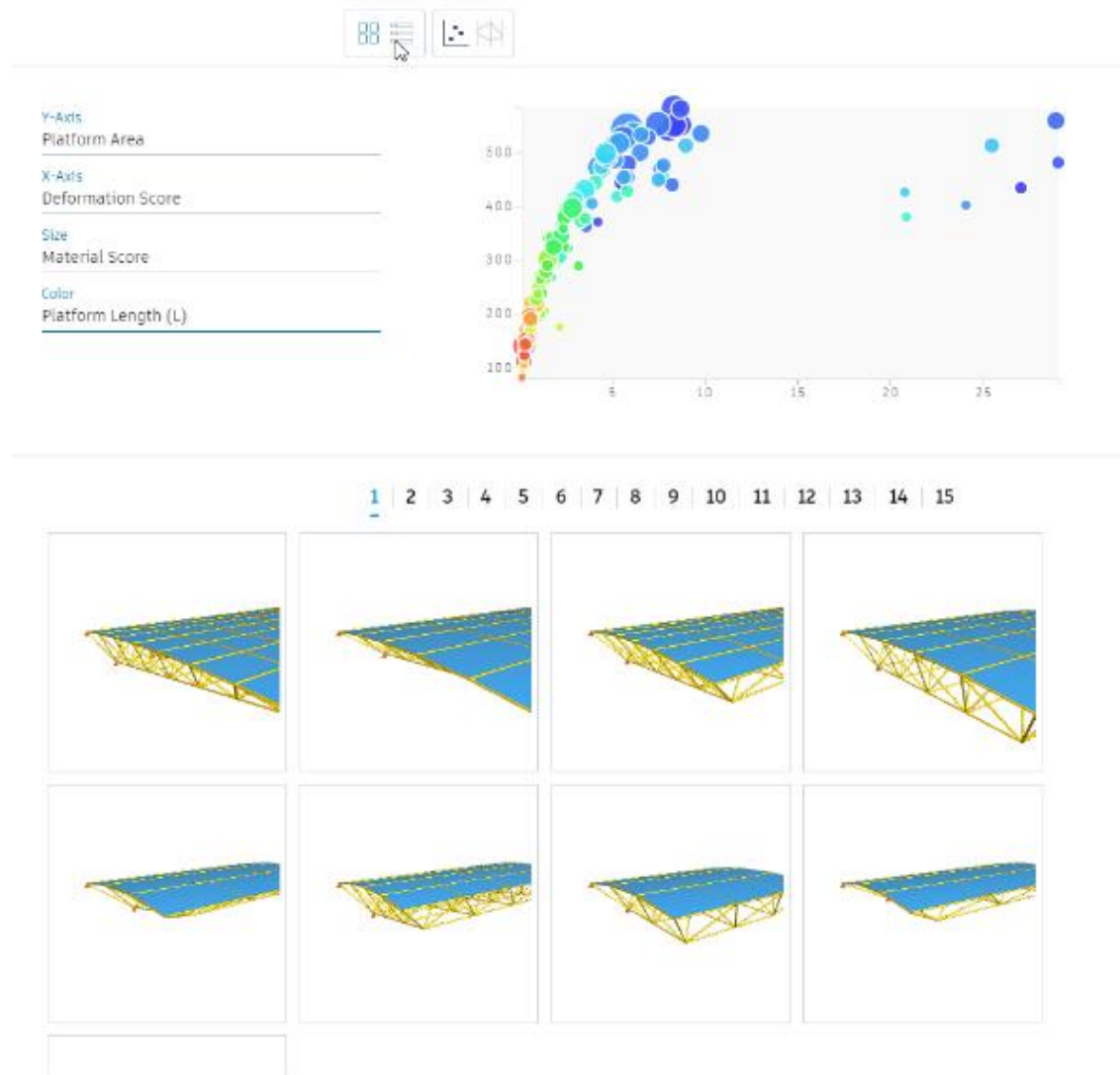
Platform Area

180

MINIMIZE

Settings

In Refinery, use the icons at the top to choose the output visualization (diagram, table, thumbnails, parallel coordinates diagram).



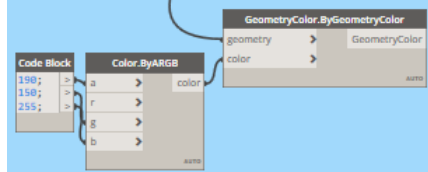
By clicking one of the thumbnails or a scatter point in the diagram or a line in the table, the input parameters in Dynamo change accordingly.

When the run method in Dynamo is set to Automatic, you can see immediate result in the Dynamo graph canvas.

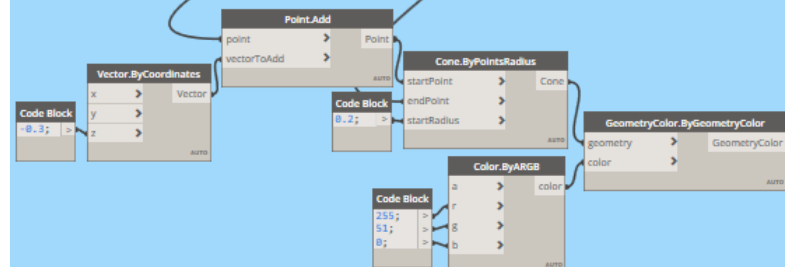
Visualization of Geometry

The blue groups in the script contain nodes to represent the generated geometry with other colors.

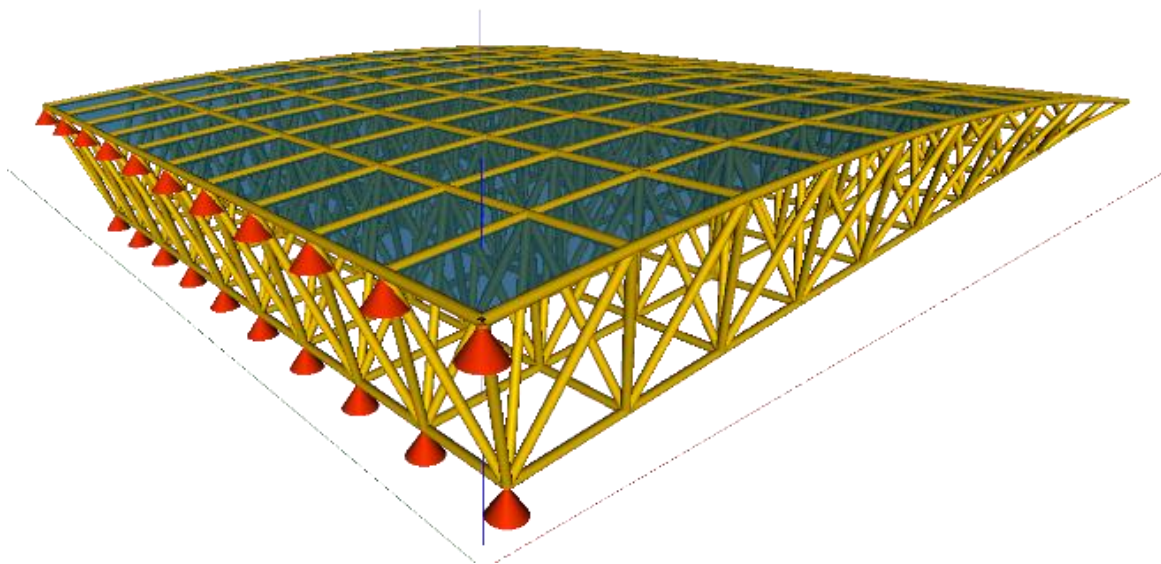
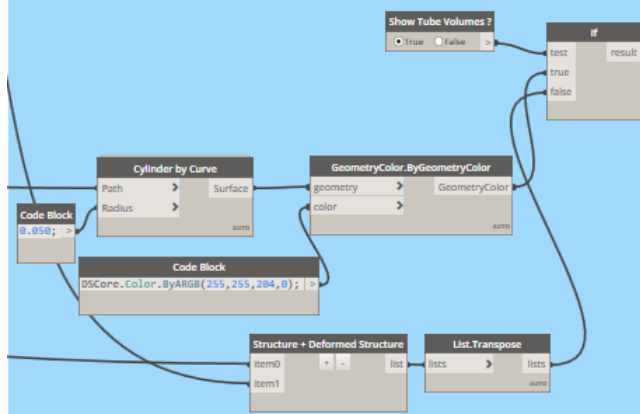
Visualize roof surface



Visualization of supports



Visualization of the structure



Truss Shape Optimization

Project Description

In this case study shape and size optimization are performed to find the best solution based on these criteria:

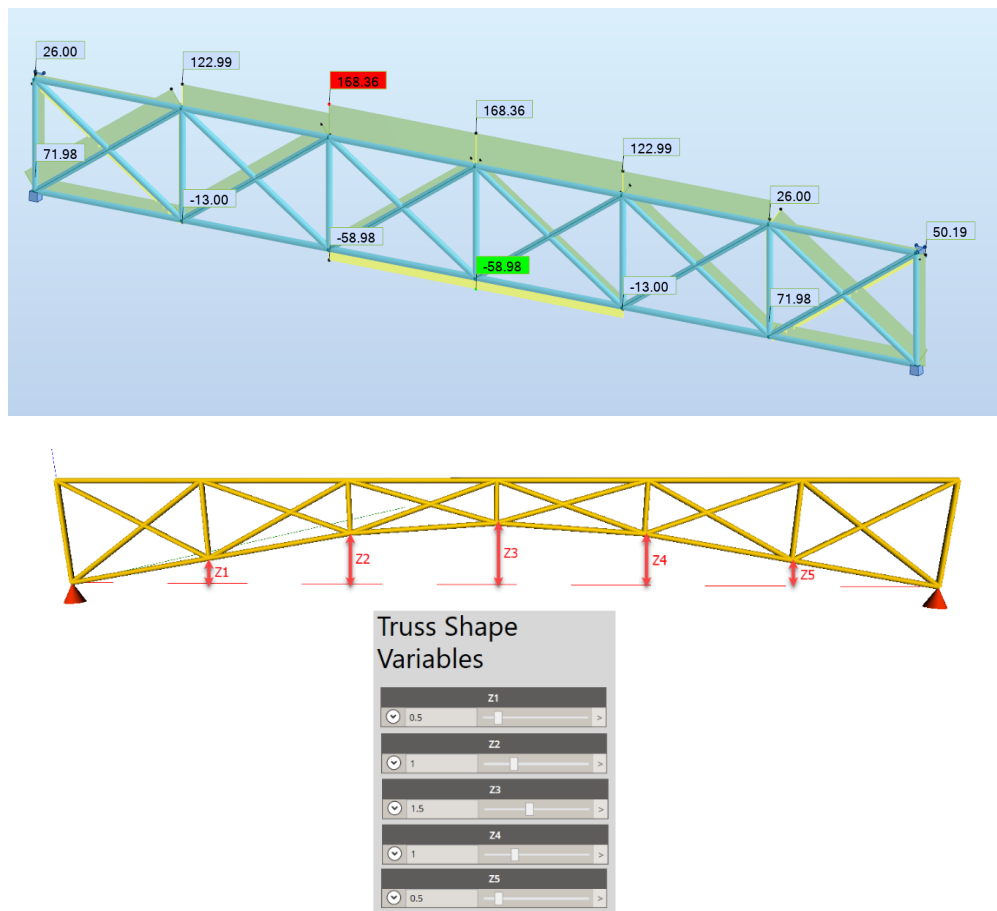
- Minimize Stress Score
- Minimize Structure Weight
- Minimize Bar Displacement

The shape of the truss can be changed, by changing the geometry inputs for each bottom truss point between the supports, given by points Z_1 to Z_{n-2}

n = number of struts

Z_0 = start node

Z_{n-1} = end node

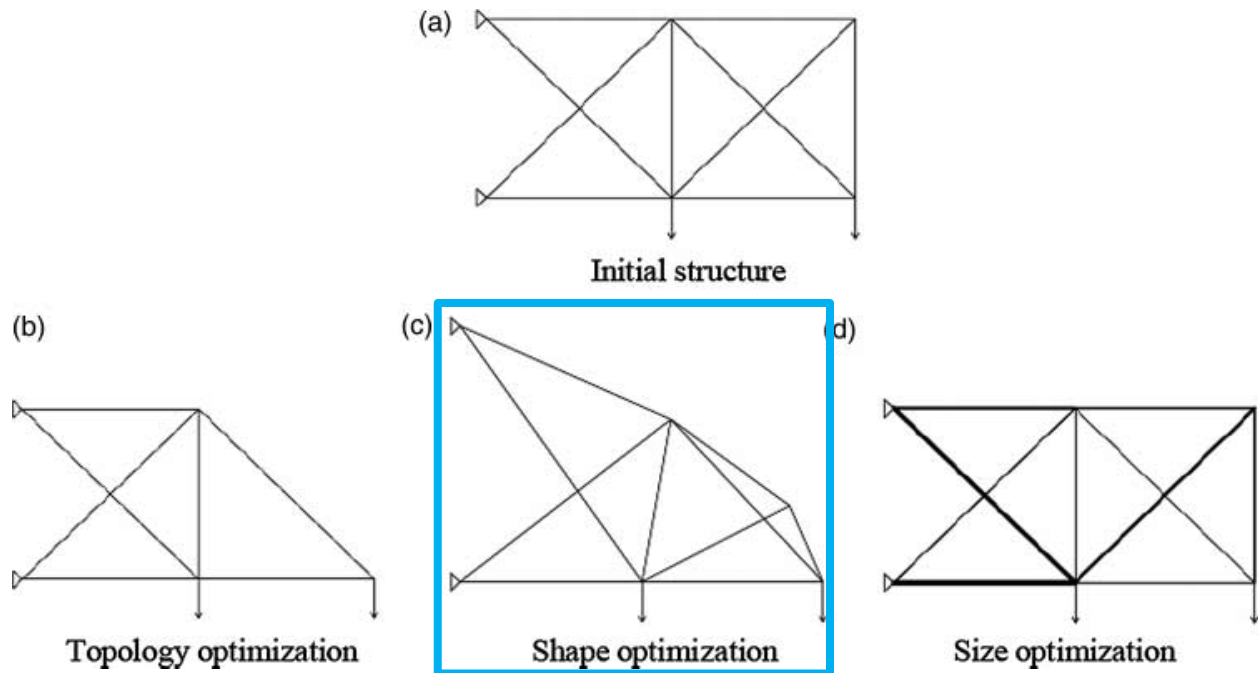


The optimization is performed with several methods:

- Design Exploration
- Brute-force search
- Genetic Algorithm Optimization

Shape Optimization

Shape optimization is part of the field of optimal control theory. The typical problem is to find the shape which is optimal in that it minimizes a certain cost functional while satisfying given constraints. In many cases, the functional being solved depends on the solution of a given partial differential equation defined on the variable domain.



Method 1 – Design Exploration

In this method the analysis is performed for a single case. The workflow created with this method can be used in custom Dynamo nodes for the Brute-force or GA Optimization method later.



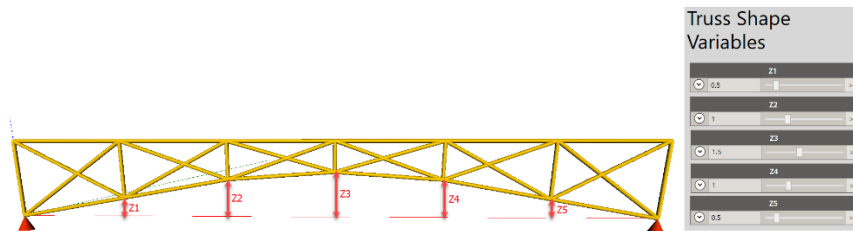
Important note

Before executing this script, you need to

1. Open Robot Structural Analysis
2. Open the file *Truss Shape Optimization Template.rtd*
3. Open the Dynamo script
4. Make sure you have the Directory folder assigned to the right one where you want to store the generated RSA models.

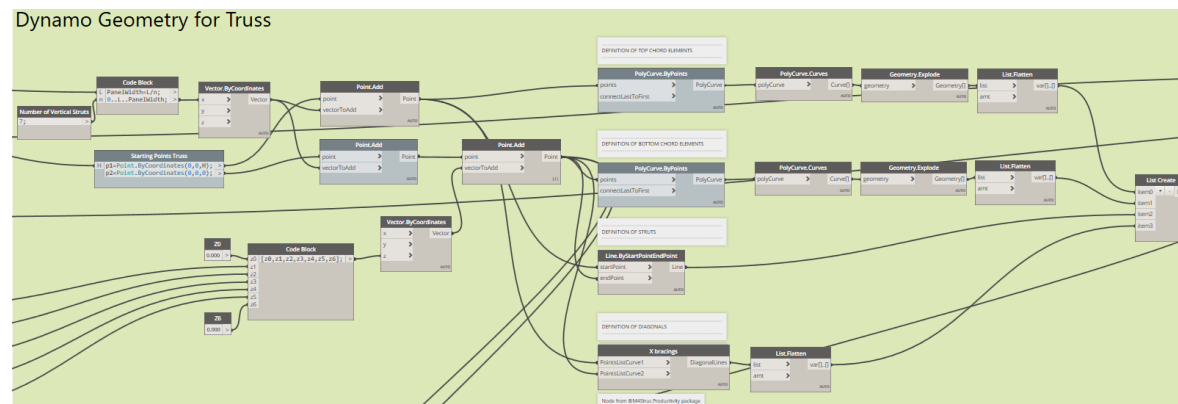
Configuration of the truss shape model

The basic layout of the truss is defined by several inputs such as length, start height, ...
The shape will be changed by driving the Z_i variables.



When the points are defined by all the variables, they are connected with each other by using *Polycurve* nodes.

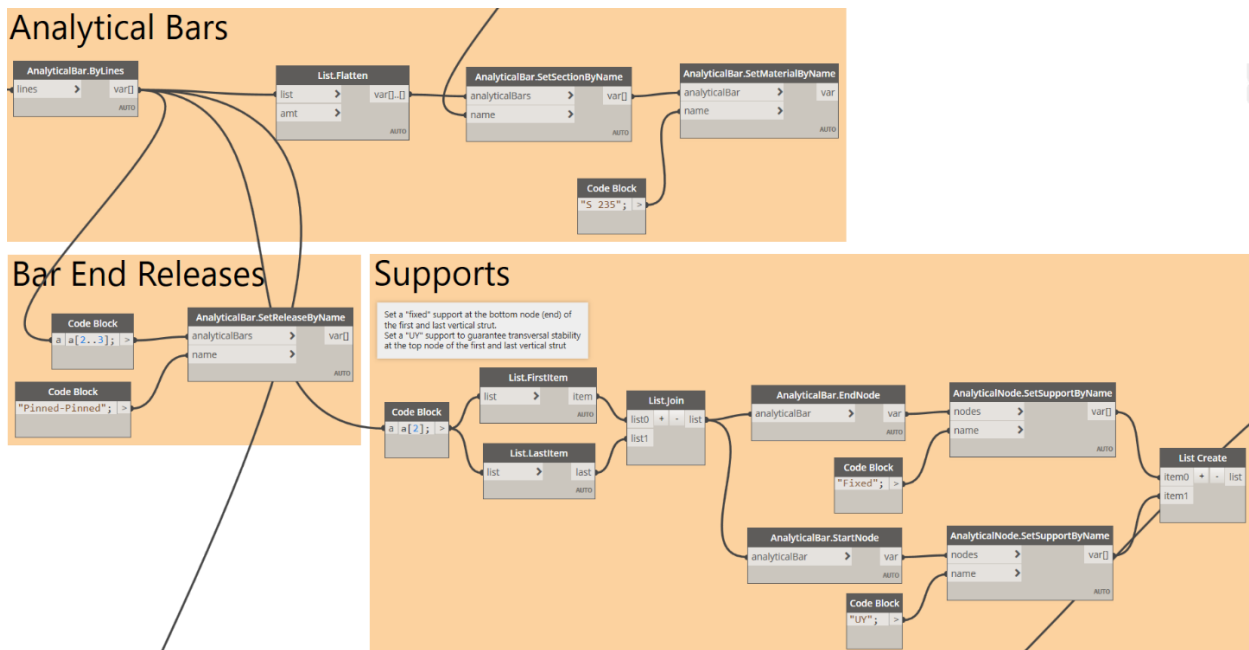
The bracings in the truss are created with the custom node *X Bracings* from the *BIM4Struc.Productivity* package.



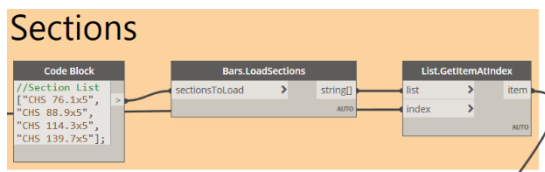
Configuration of the analysis model

The analysis is performed in Robot Structural Analysis 2019 (RSA). To connect Dynamo with RSA you need the package *Structural Analysis for Dynamo*. With this package you can create geometry and assign simulation criteria based on Dynamo geometrical input.

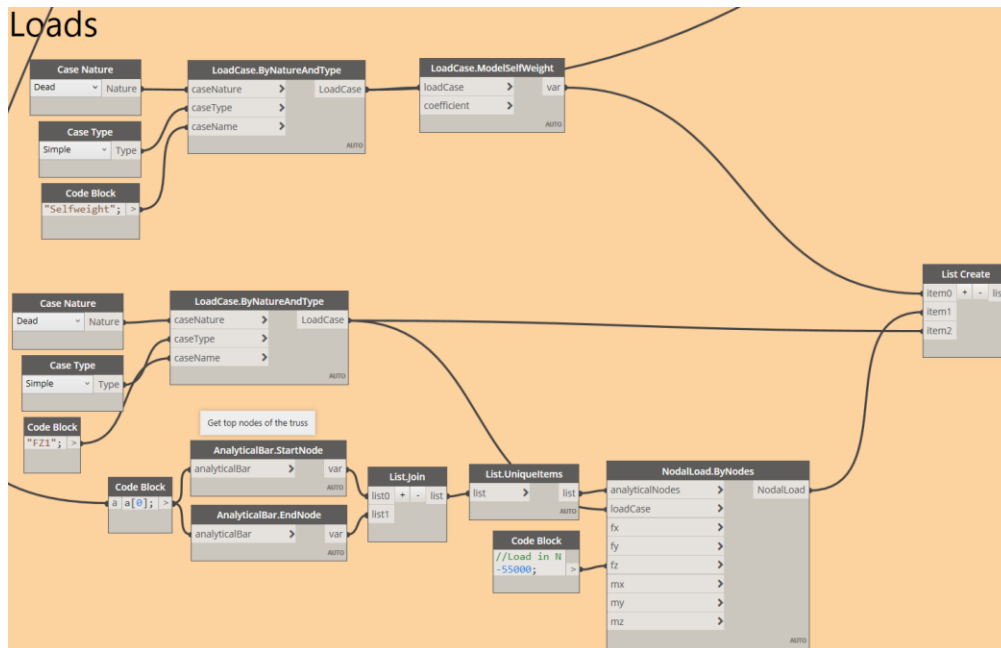
1. Assign the **lines** from the Dynamo geometry to the *AnalyticalBar* nodes. Don't connect the outputs of the polycurves, as they will not be accepted.
2. Set the bar end releases for the generated analytical bars
3. Set the supports to the end nodes. Make sure the support name exists in the open session of RSA.



4. Set the section with the *SetSectionByName* node. The input needs to be a string and the section need to be loaded in the open session of RSA.

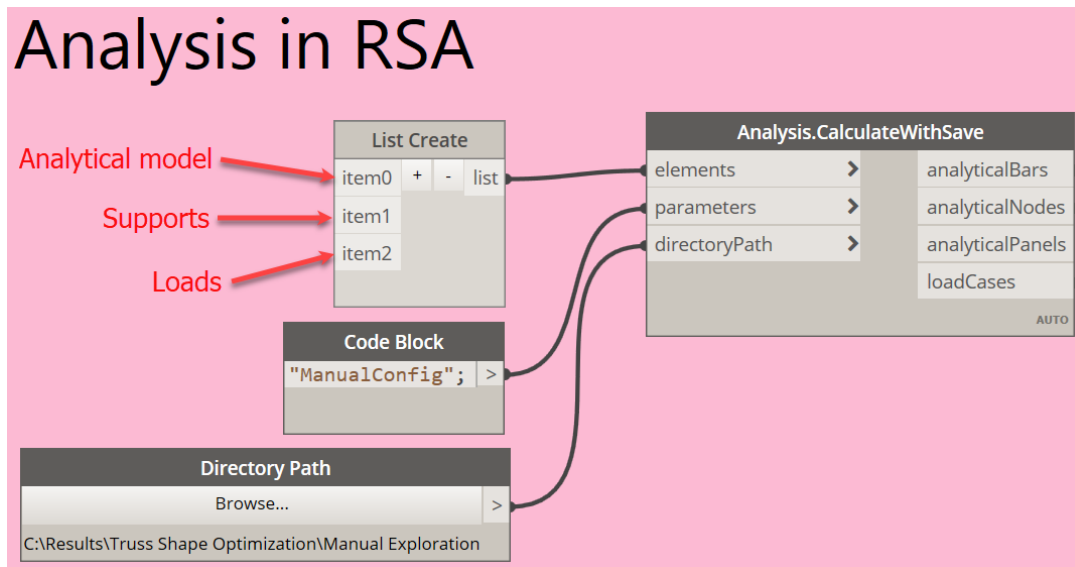


5. Apply the loads on the appropriate analytical bars and nodes. The loads are expressed in Newton in this example.

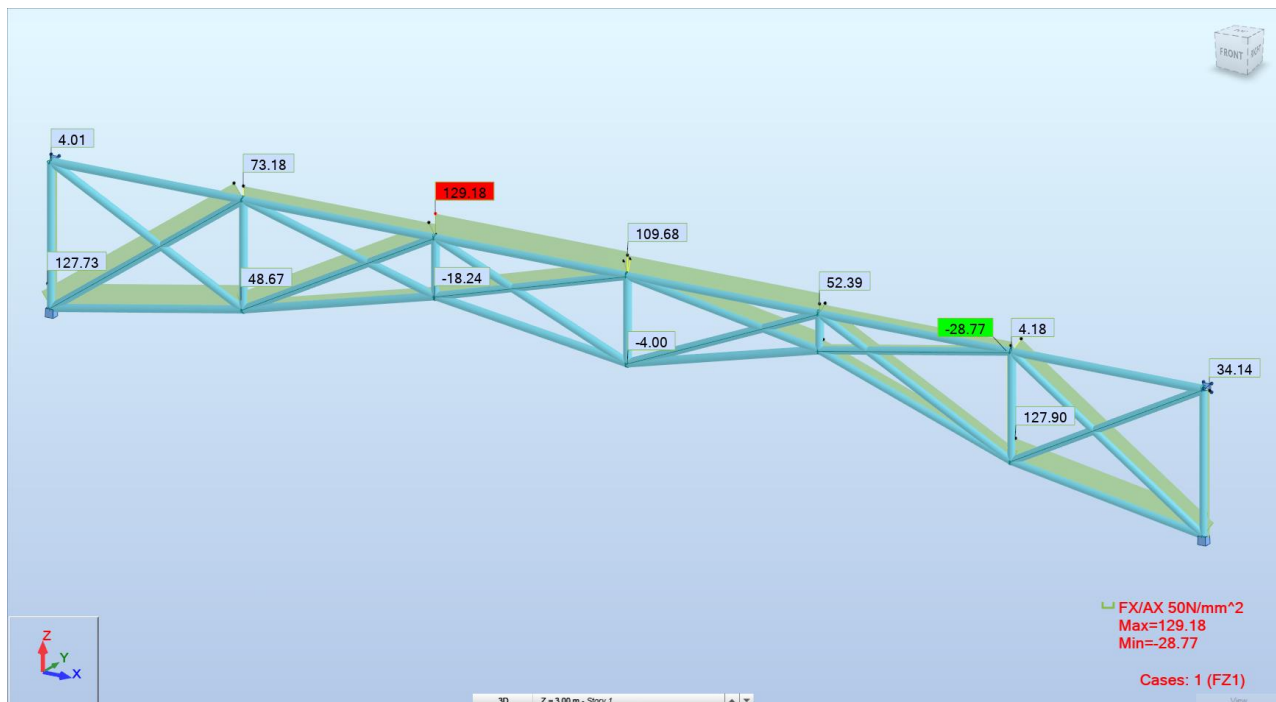


Perform the structural analysis

The analysis in RSA is driven by Dynamo and the execution is ensured with the *Analysis.CalculateWithSave* node. This node reads the input for the analytical model, supports and loads, performs the analysis, saves the model on a given location (at the input group of the script) and returns results for the analytical bars and nodes.

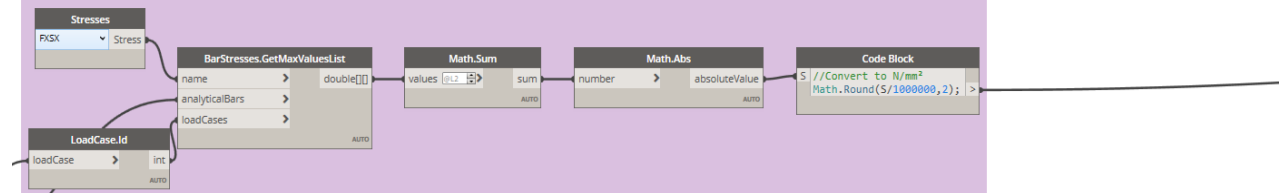


The results can be verified in RSA. Click on Results > Diagrams for Bars... and choose for the result you want to verify.

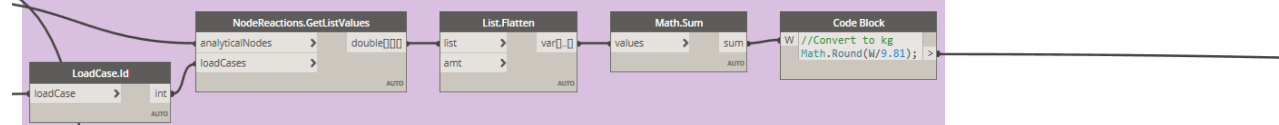


The package has dedicated nodes to extract the results from the analysis. Make sure you assign the right load case ID for the results extraction.
In this case, the total stresses resulting from the selfweight and the nodal forces are considered. Similar is done for the bar displacement.

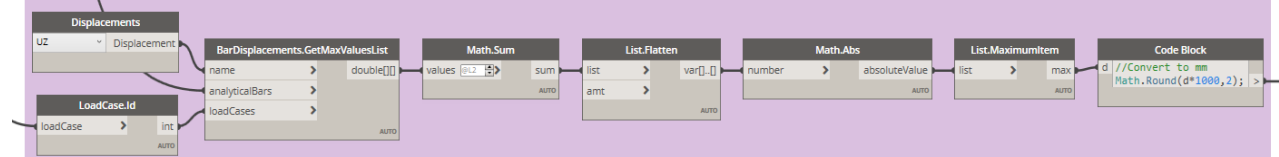
Extract Stresses



Extract Total Selfweight



Extract Deformation for all cases



Evaluate results

Before the results are exposed in the watch nodes, the stresses are evaluated. In optimization problems it can be useful to assign scores to designs, to find the optimal result more efficient.

The scoring procedure is done with a Python script.

```
import clr
clr.AddReference('ProtoGeometry')
from Autodesk.DesignScript.Geometry import *
#The inputs to this node will be stored as a list in the IN variables.
dataEnteringNode = IN

#Actual stress in the bar
ActualStress=IN[0]
#Set the maximal allowable stress
AllowableStress=IN[1]
#Sets the scoring system for the stress ratios
ScoreMinStress=IN[2]
ScoreMaxStress=IN[3]
#Ratio range for optimal design
MinRatio=IN[4]
MaxRatio=IN[5]

Score=0
o=0
u=0

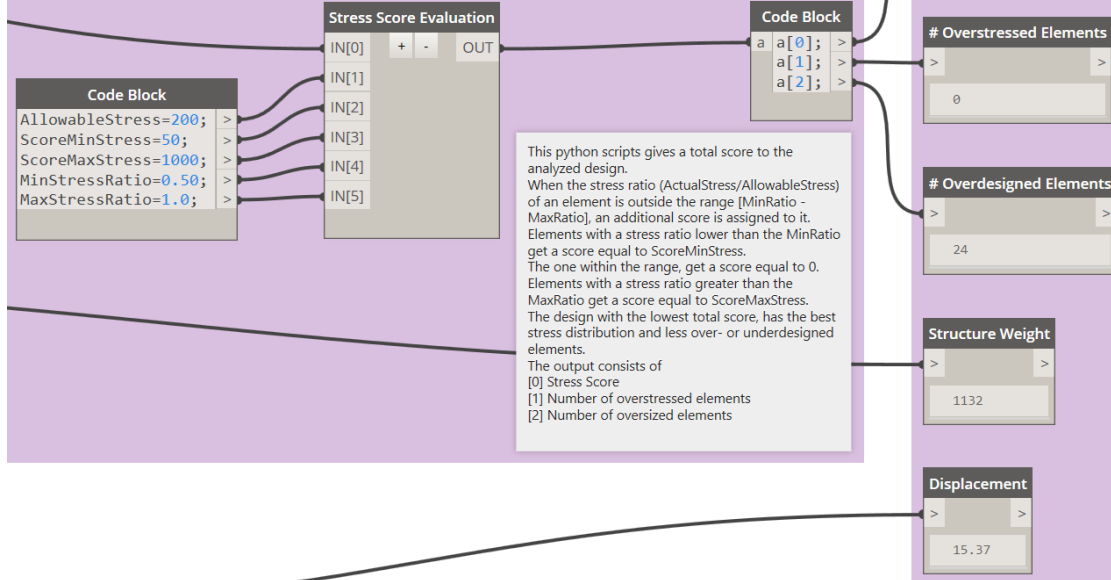
for s in ActualStress:
    StressRatio = s / AllowableStress
    if StressRatio >= MaxRatio:
        Score=Score+ScoreMaxStress
        o=o+1
    if StressRatio < MinRatio:
        Score=Score+ScoreMinStress
        u=u+1

#Assign your output to the OUT variable.
OUT = Score, o, u
```

The scoring system is based on the principle below. Depending on the stress within a bar element, a specific score is given. The total of all element scores returns the Stress Score of the design option.

		Elements per domain	Stress Score per domain	Stress Score per domain	
Overdesigned	$R < 0.5$	6	5	30	
Within Stress Ratio Domain	$0.5 < R < 1.0$	10	0	0	
Overstressed	$R \geq 1.0$	5	1000	5000	Stress Score = 5030

Evaluate Stresses



Method 2 – Brute Force Search

In this method the analysis performed in Method 1 is executed and evaluated for multiple values of the input variables. This is done in an iterative way. Each design option is analyzed and evaluated. The results are collected by the analysis function for later evaluation processing.



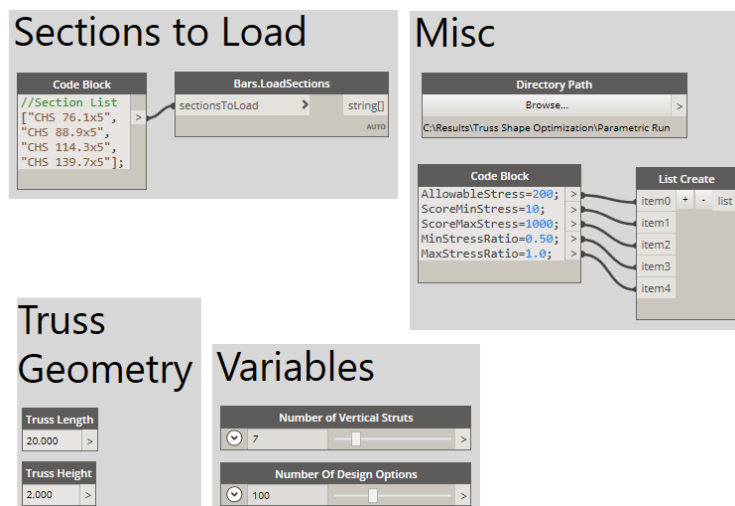
Important note

Before executing this script, you need to

1. Open Robot Structural Analysis
2. Open the file *Truss Shape Optimization Template.rtd*
3. Open the Dynamo script
4. Make sure you have the Directory folder assigned to the right one where you want to store the generated RSA models.

Design Space Variable Domain

The variables of this optimization process are either constant or driven by the brute-force process.



Sections to load:

This group contains strings of the profile sections that need to be loaded in RSA before running the iterations.

Misc.:

Contains nodes for the assignment of the folder where the RSA designs need to be stored and

a code block that defines the constraints for the stress score evaluation. These values can be different for your own evaluation process.

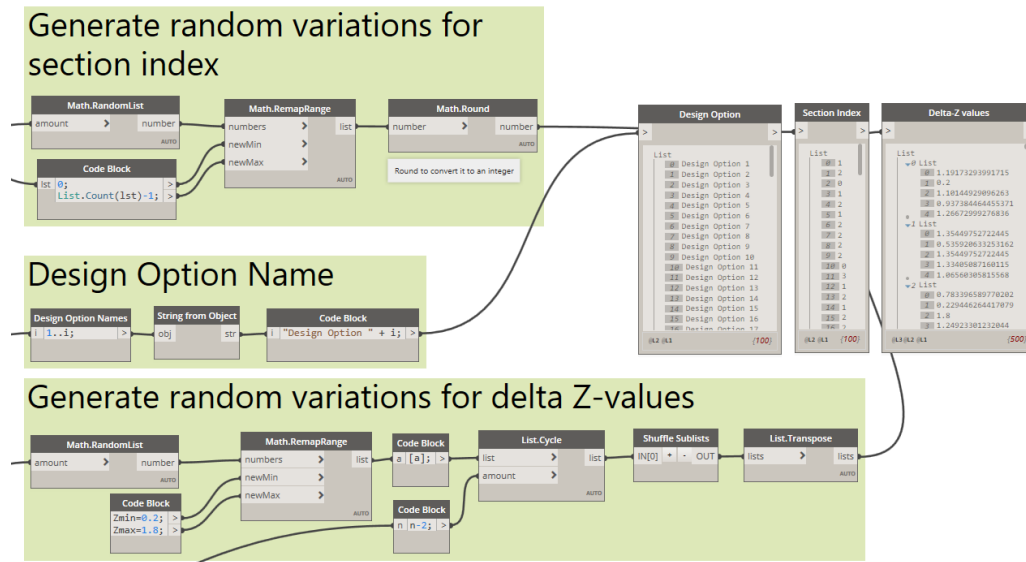
Truss Geometry

These are constant values defining the truss boundaries.

Variables

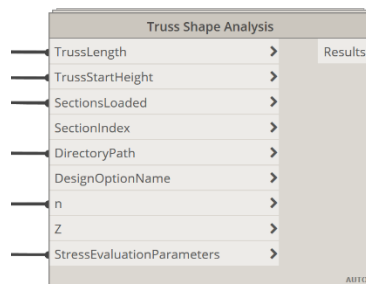
These variables are driving the brute-force optimization inputs. The *Number of Design Options* defines how many iterations are done.

The values for the section index are randomized. Also, the values for the range of delta-Z values (Z_1 to Z_{n-2}) are randomized and shuffled for each design option.



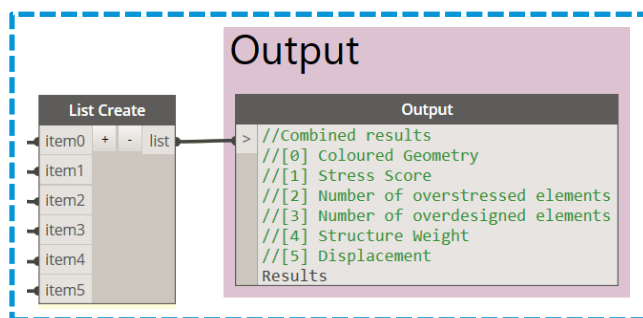
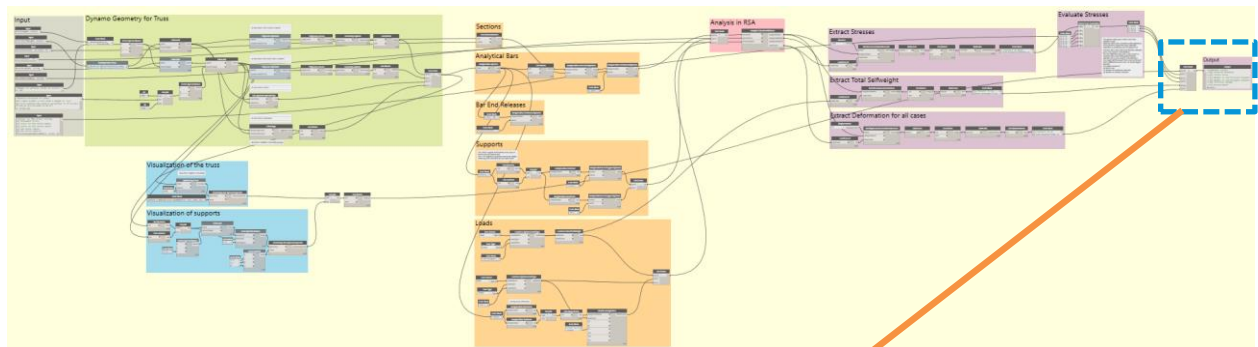
Truss Shape Analysis Function Definition

To run the analysis for multiple input variables it's necessary to create a custom node from the design and analysis process. This is stored in the *Truss Shape Analysis.dyf* custom node. After loading the script, you will find it in the BIM4Struc.Projects/Truss Shape Analysis category.



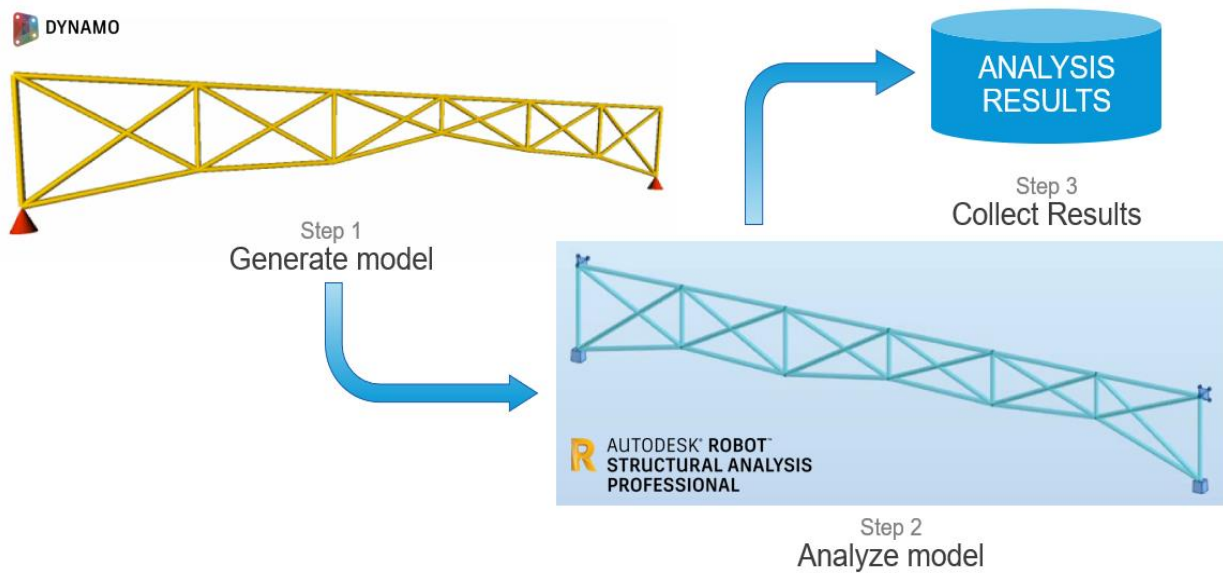
This custom node contains everything you need to generate and analyze truss according to the "Design Exploration" method.

The output of the custom node contains the geometry and all the results needed for optimization.



Brute-Force Search Execution

When the script is executed, each iteration the geometry model in Dynamo is modified and a new RSA file is created according to the new design option. All the results are collected in the output of the custom node for further evaluation.



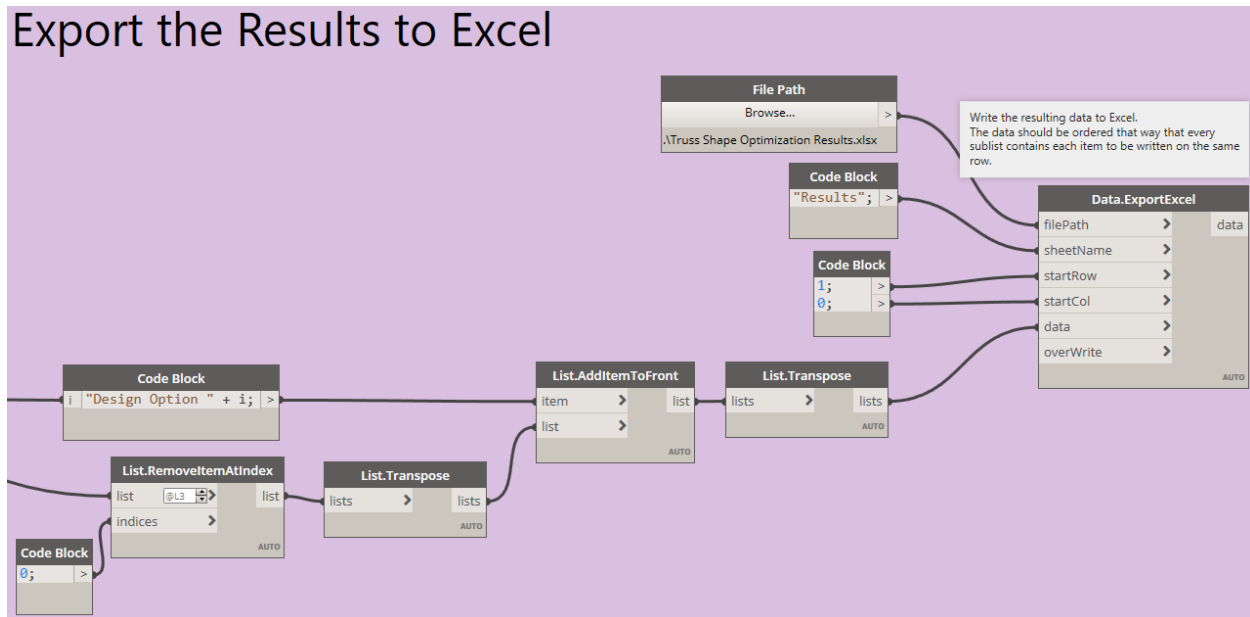
Evaluation of Results

After all iterations have been executed, the results can be visualized in several ways.

Export to Excel

With the *Data.ExportExcel* node the results can be exported to a pre-defined Excel sheet. Make sure you point to the right file path for the sheet delivered with the datasets.

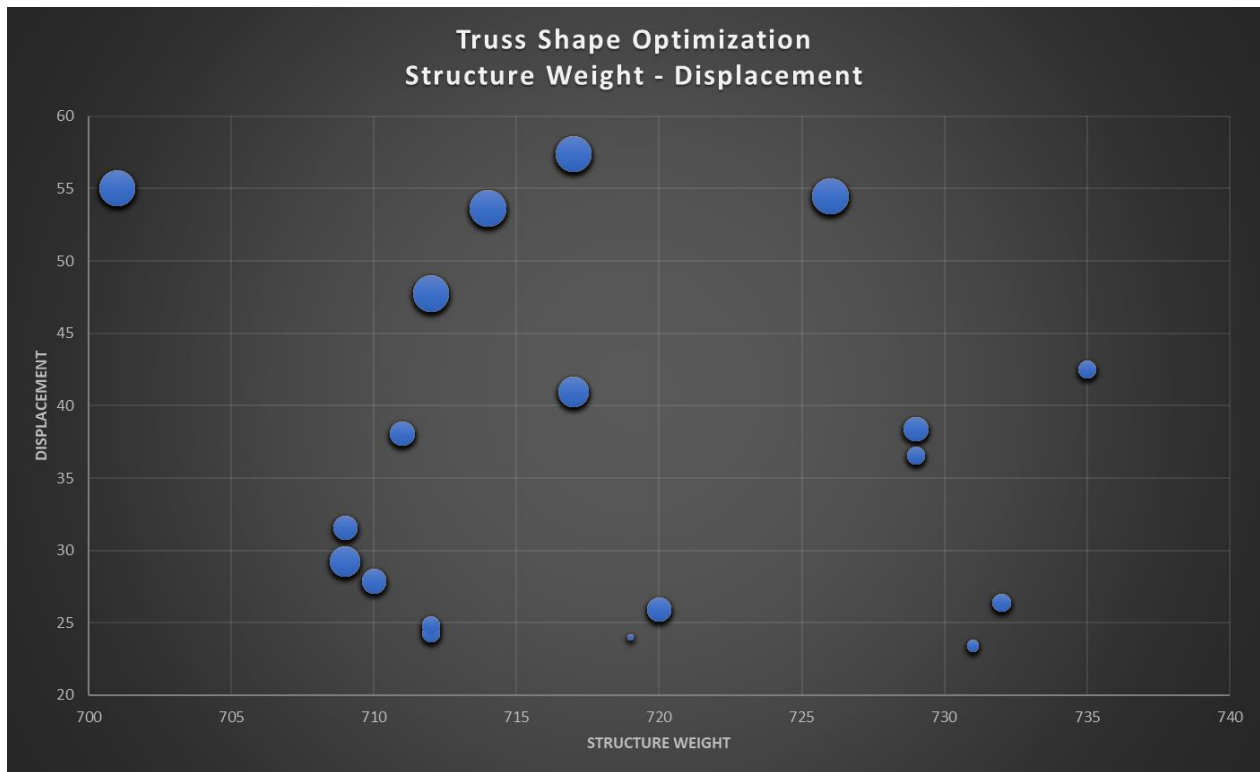
The data needs to be stored in rows, so therefore all results and input values are changed from columns to rows with the *List.Transpose* node.



More information on how to export to Excel can be found here:

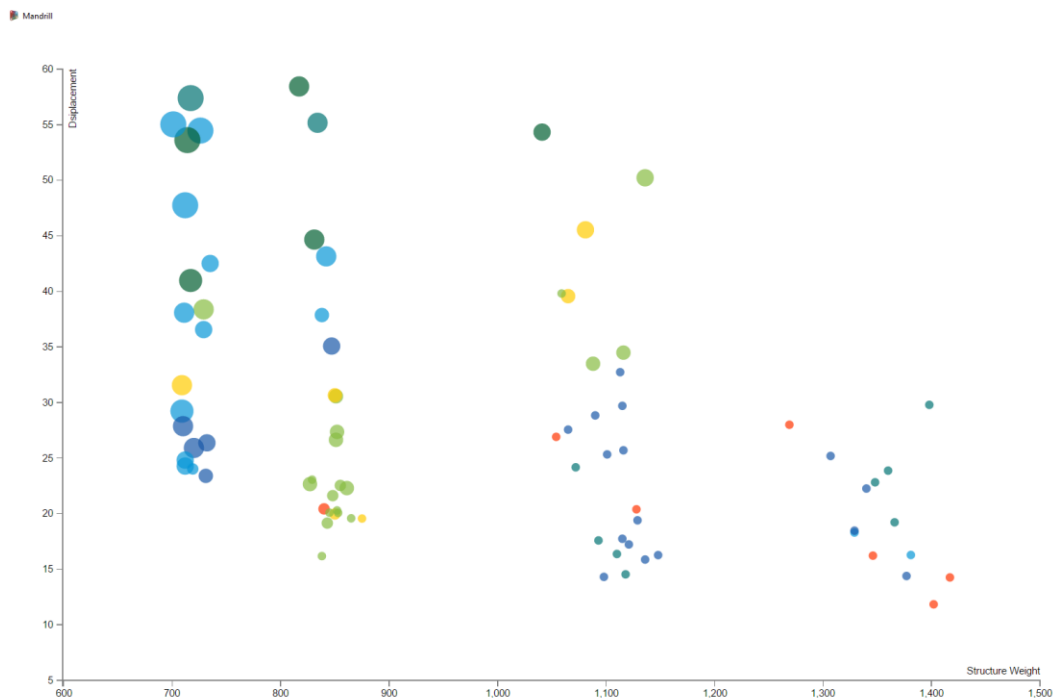
<http://dictionary.dynamobim.com/#!/Office/Excel/Action/WriteToFile>

Design Option	Stress Score	Number of Overstressed Elements	Number of Overdesigned Elements	Structure Weight	Displacement
Design Option 1	1050	0	21	870	21.08
Design Option 2	3150	2	23	1059	34.53
Design Option 3	2250	1	25	1146	32.09
Design Option 4	1100	0	22	714	23.91
Design Option 5	1250	0	25	1118	19.55
Design Option 6	1250	0	25	1123	15.66
Design Option 7	4050	3	21	728	24.17
Design Option 8	5850	5	17	716	40.81
Design Option 9	1050	0	21	1365	48.73
Design Option 10	1250	0	25	1088	16.75
Design Option 11	4950	4	19	824	33.93



Visualize with Mandrill

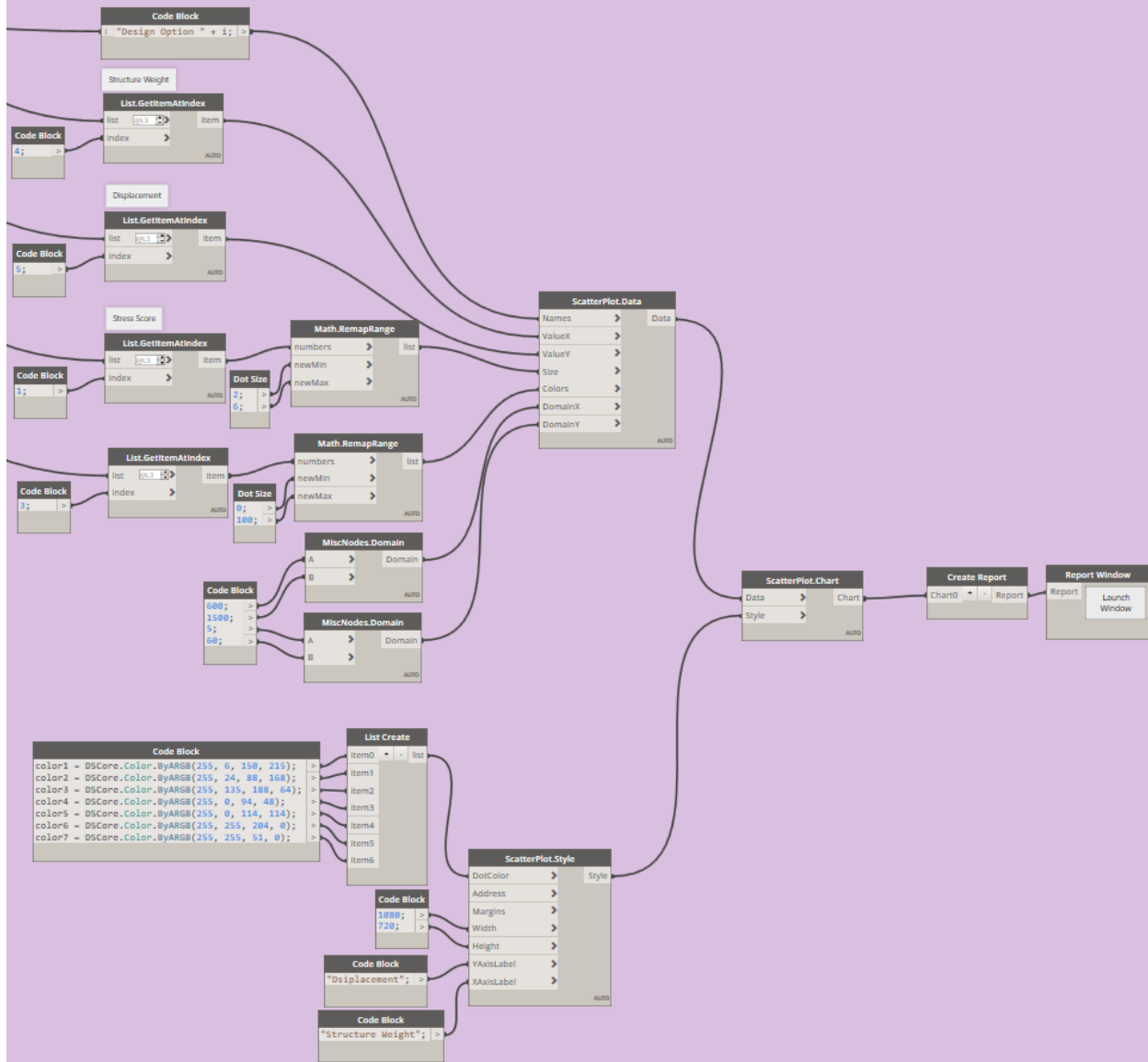
The Mandrill package contains nodes to visualize data in Dynamo. In this case a scattered plot is used where the points are plotted in a X-Y diagram. The size and color of the plot point can also be variable depending on data.



In this script the data is visualized as follows:

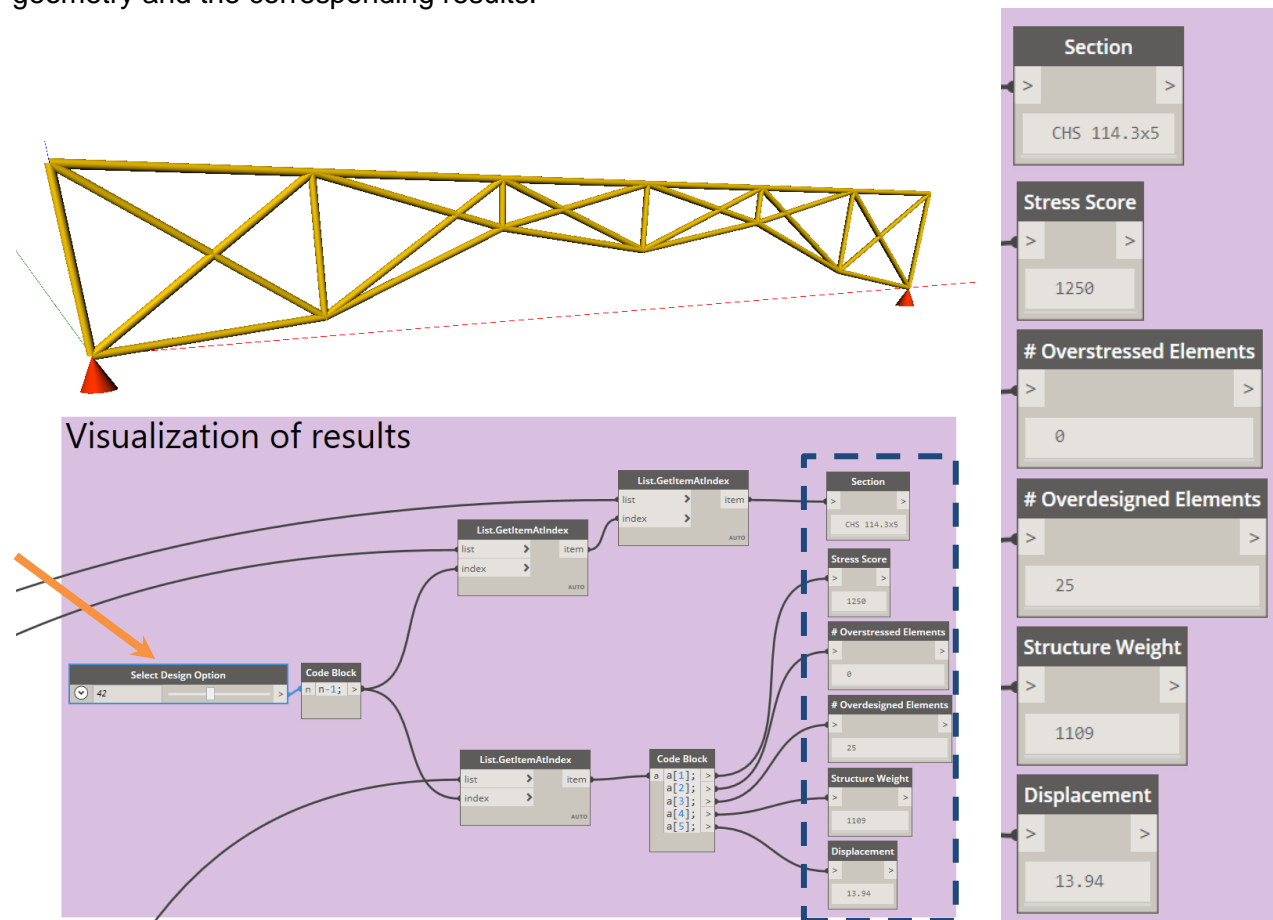
- X = Structure Weight
- Y = Displacement
- Dot size = Stress Score
- Dot color = Number of oversized elements

Visualize Results with Mandrill



Visualization of design options

Each design option can be visualized separately in this part below of the script. It shows the geometry and the corresponding results.



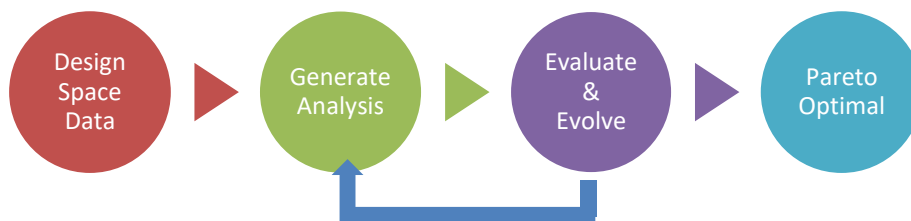
Every run of the custom node generates a new RSA file in the specified folder. This allows the user to review the design option in RSA for further analysis.

Name	Date modified	Type	Size
Design Option 1_124058.rtd	06/11/2018 12:40	Autodesk RTD Do...	1 008 KB
Design Option 2_124101.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB
Design Option 3_124105.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB
Design Option 4_124108.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB
Design Option 5_124112.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB
Design Option 6_124116.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB
Design Option 7_124120.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB
Design Option 8_124125.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB
Design Option 9_124129.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB
Design Option 10_124134.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB
Design Option 11_124139.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB
Design Option 12_124145.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB
Design Option 13_124150.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB
Design Option 14_124156.rtd	06/11/2018 12:41	Autodesk RTD Do...	1 028 KB

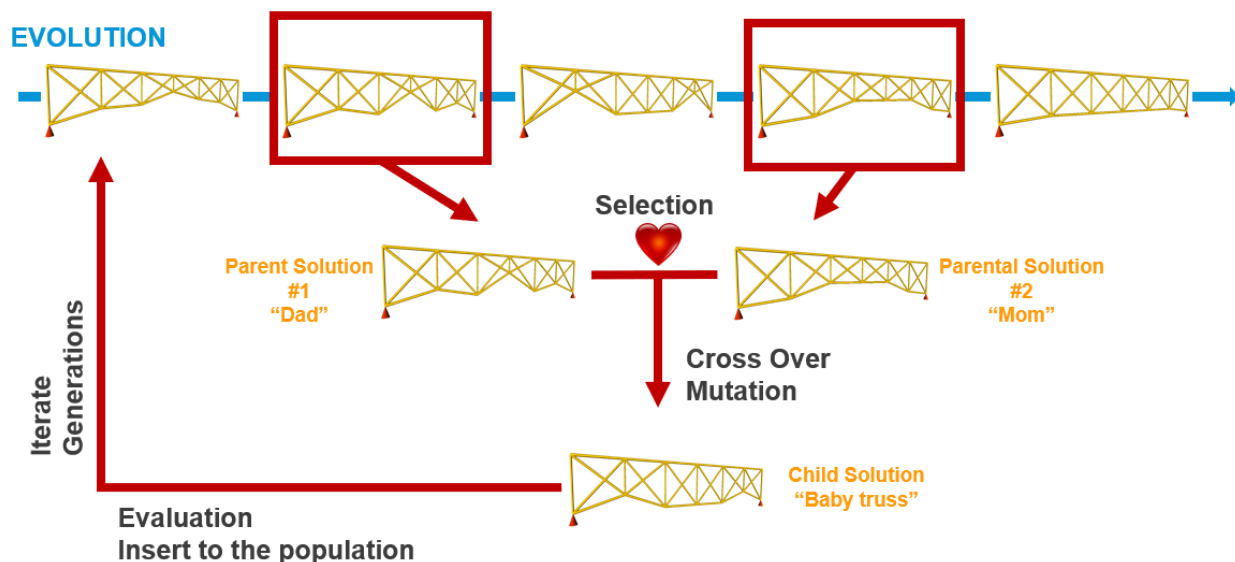
Genetic Algorithm Optimization

The “brute-force search” method is very time consuming, as it considers every design option, even if the design option results into stress ratios that are beyond the allowable stress ratio range (read: “excessive stress score”). Besides that, the brute-force method as used above will only calculate the given set of combinations and won’t find alternatives that might be better.

To speed this up, and to find more accurate results we need to introduce Genetic Algorithms in the analysis process. The GA Optimization will use specific methods making a natural selection of the initial solution list and by using cross-over and mutation algorithms to find the optimal solution. Compare it with “introducing the natural evolution theory of Darwin” into computational design. The main parts of the script that has been made in Method 2, can be reused by adding specific nodes from the Optimo for Dynamo package. Read more about the use of this package on [this link](#).



Genetic Algorithm Optimization with Optimo



Based on the fitness values, a GA performs a selection operation that reproduces a set of solutions with higher fitness values from the previous generation to fill a mating pool. A crossover operation is then pursued with which two parent solutions in the mating pool are randomly selected and interchange, with a prescribed crossover probability, their respective string components at randomly selected bit locations referred to as cross sites. The resulting new solutions are called children or offspring. This step is meant to hopefully combine better

attributes from the parent solutions so that child solutions with improved merits could be created. The next operation in GA is mutation that changes the genotype value at one or more randomly selected bit locations in a child solution with a prescribed mutation probability. This operation serves to possibly recover useful information that may never be accessible through selection and crossover operations and therefore encourages search into a completely new solution space. After these three basic operations, a new generation is created. The search process continues until prescribed stopping criteria are met, for example, when the maximum generation number is reached, or improvement of the generation-wise optimized solutions is negligible.

Unlike single-objective problems where the objective function itself may be used as the fitness measure (possibly further with scaling and constraint-handling treatment), a multi-objective optimization algorithm needs a single fitness measure that reflects the overall merit of multiple objectives.

Important note

Before executing this script, you need to

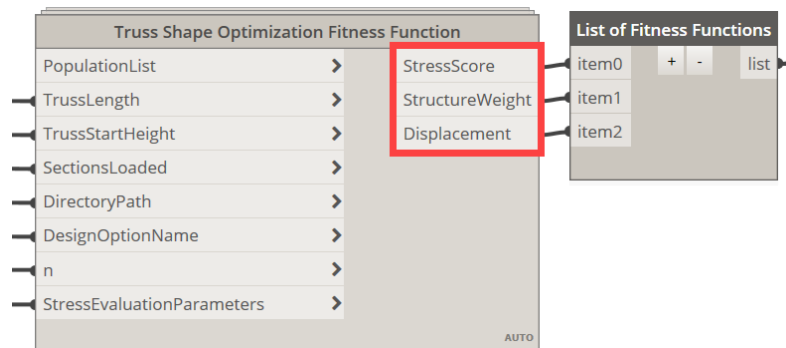
1. Open Robot Structural Analysis
2. Open the file *Truss Shape Optimization Template.rtd*
3. Open the Dynamo script
4. Make sure you have the Directory folder assigned to the right one where you want to store the generated RSA models.

Representation of Optimization Problem

Before we start with building up the script, it is important to first represent the optimization problem which consists of an “objective”, “design variables” or “population”, “design variables domain” and finally the design “constraints”.

Objective

With the optimization we want to find a solution with minimized structure weight, stress ratio and displacement. This means we have a triple objective. This could be done with a MOO (Multi-Objective Optimization).



Design Variables

The design variables for this optimization are expressed with three parameters:

- Section Index (defines the cross-section type that can be used, by referring to the section list using an index).
- Delta-Z values (Z_1 to Z_{n-2}): the values driving the bottom segments of the truss

These are thus the variables that may be changed by the routine to get to an optimal solution.

Design Variables Domain

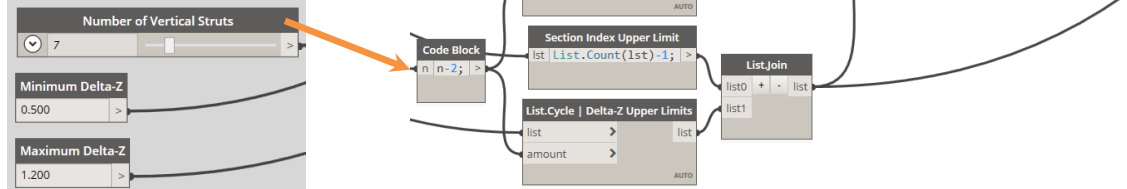
We need to give some restrictions for the design variables and give the optimization routine a boundary within to search an optimal solution for. In this case the domains are set as follows:

- $0 \leq \text{Section Index} \leq \text{Total amount of Sections} - 1$
- $0.5 \leq Z \leq 1.2$

Sections to Load



Variables



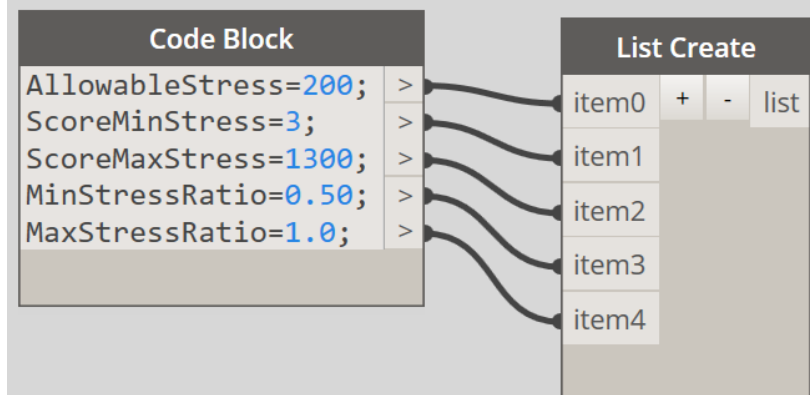
Constraints

As we don't want to have solutions with a minimal weight but with overstressed members, we need to introduce constraints. These constraints consist of some rules that will limit the usage of some design variables.

In this script final constraints are:

- MinimumRatio
- MaximumRatio

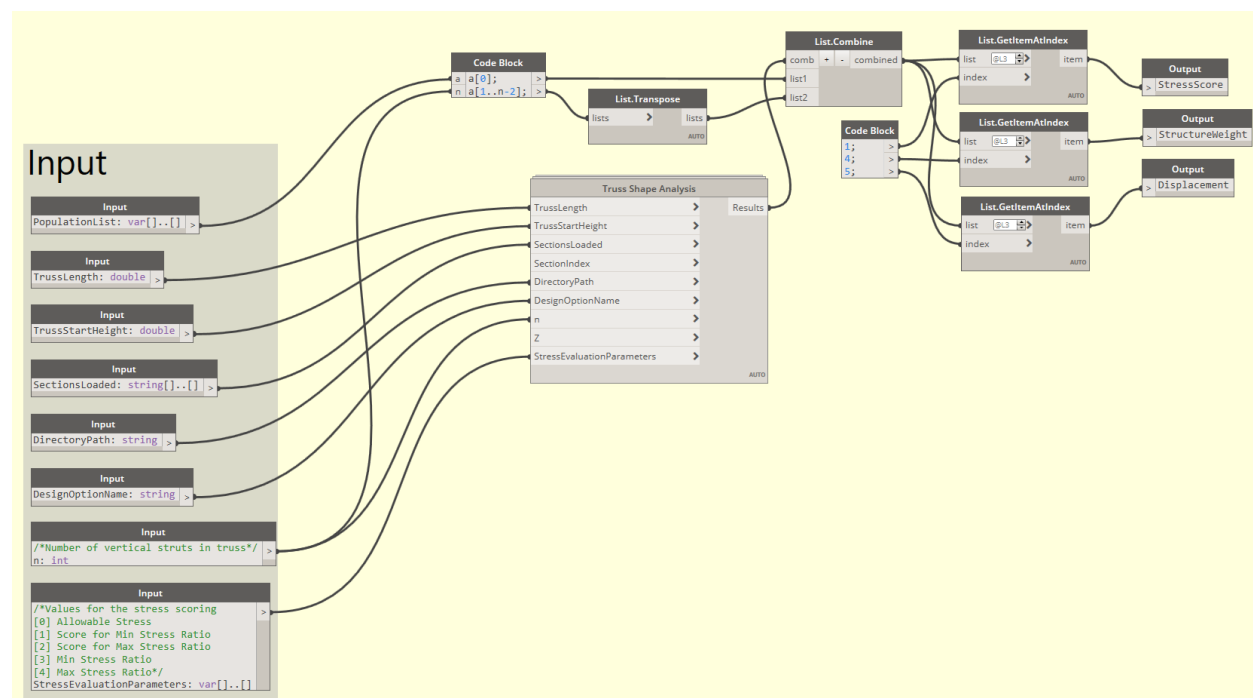
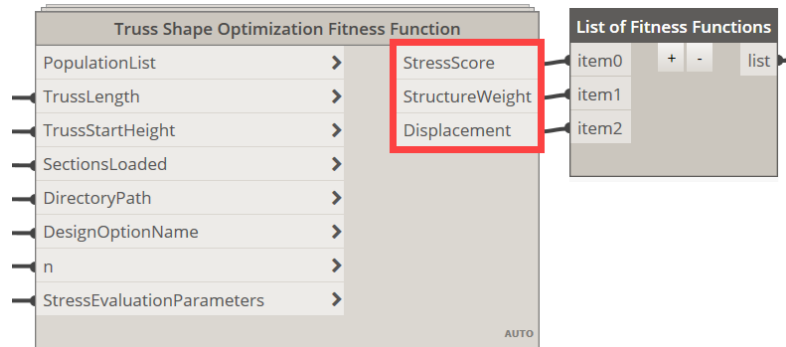
These constraints are at their way defined by the *AllowableStress* factor. A score is assigned depending on the stress ratio of an element. Read more in [this chapter](#).



Truss Shape Fitness Function

Truss Shape Fitness Function.dyf is a custom node that is designed that way to read the population list variables (in this case a list of two variables: Section Index, and Delta-Z values). Each of these lists contains a given amount of randomized values (amount is equal to the population size). The custom node contains the *Truss Shape Analysis.dyf* custom node, which is the backbone of this calculation and is described [here](#) already.

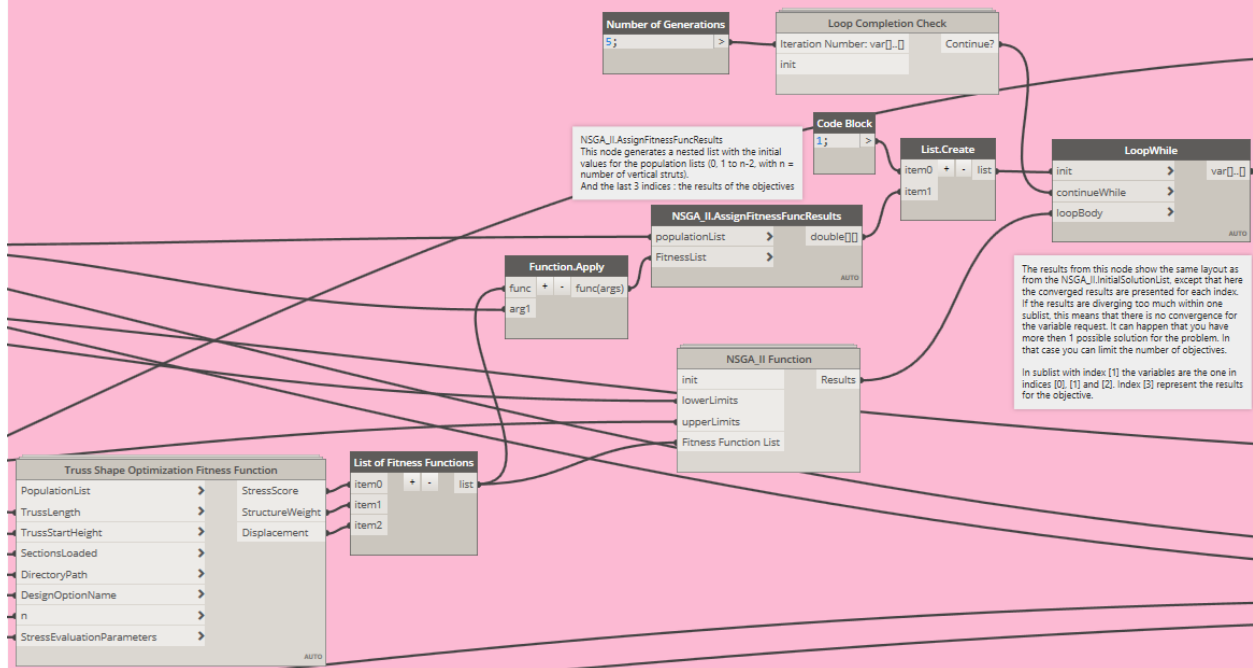
The *List of Fitness Functions* node represents the number of objectives to be optimized.



Run GA Optimization

The script is performed with the Optimo package.

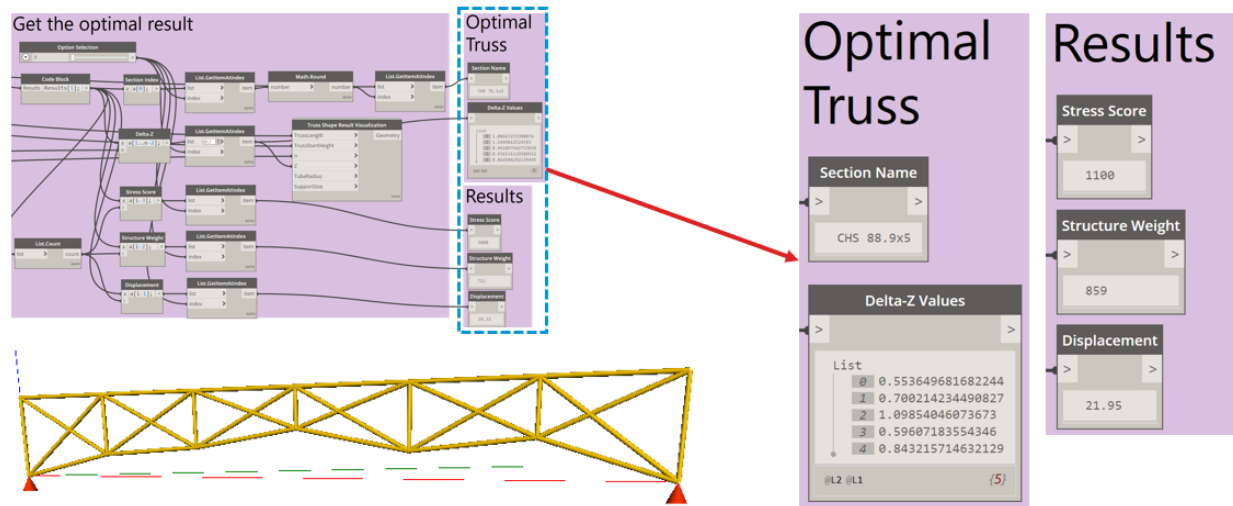
Genetic Algorithm Optimatizion



More information about the nodes used for the optimization can be found in the handout from this class: [Autodesk University 2015 – Dynam\(o\)ite Your Design for Engineers](#).

Evaluation of results

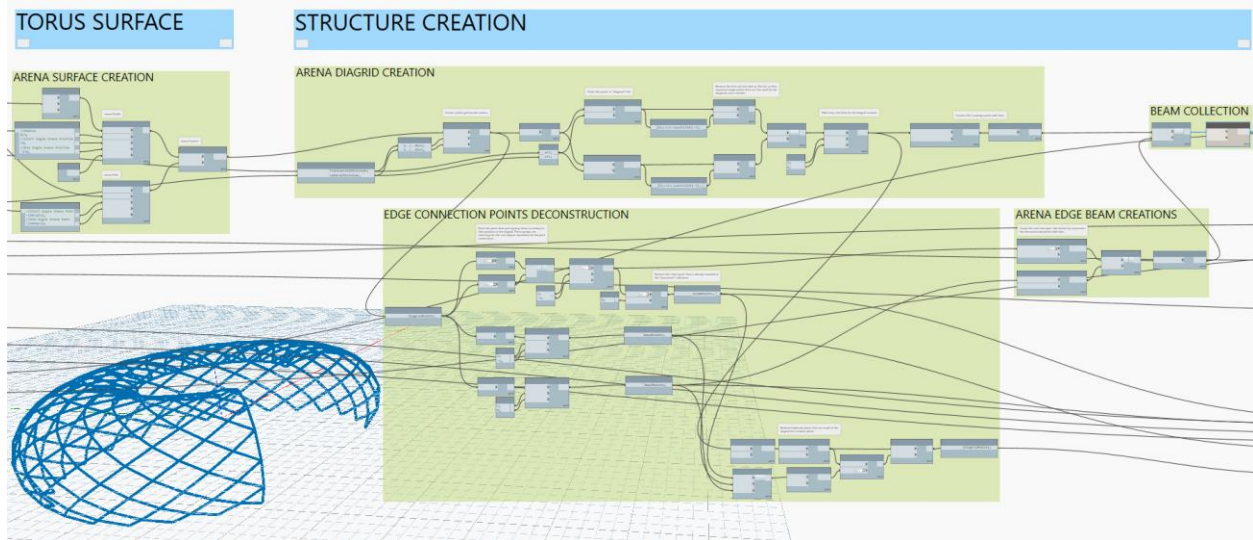
The Optimo results need to be broken down to get more details about the analysis.



Mars Torus Arena

Computational Model

First, a computational model is setup by means of nodes and custom nodes. This will create the framework for the geometry that needs to be evaluated.



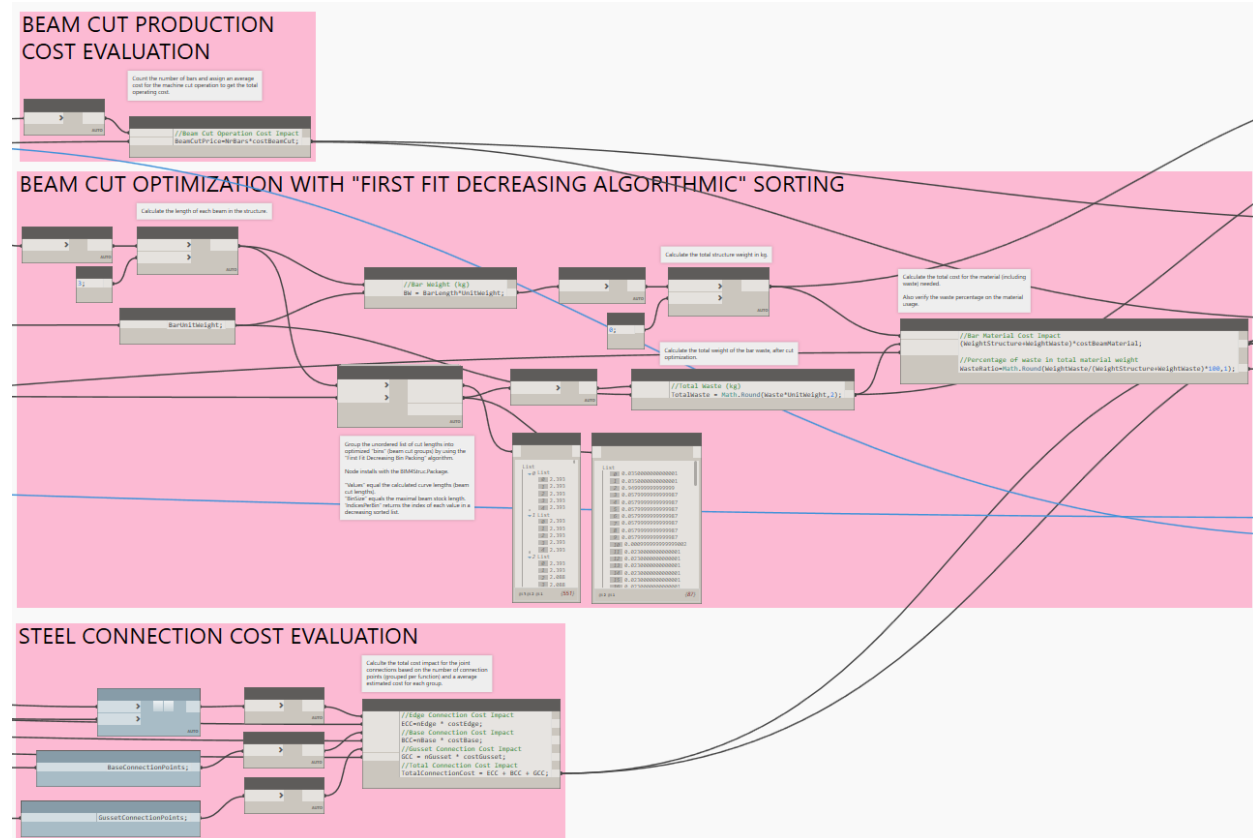
More information on how the geometry is created can be found on these AU Classes:

- *ES119852 - Dynam(o)ite Your Steel Design @ AU Las Vegas 2017*
- *Optimization of Structural Designs with Dynamo @ AU London 2018*

At the end of this handout you'll find download links for these classes.

Structure Evaluation

The generated structure is evaluated on constructional and cost objectives.



The evaluation is done for multiple topics:

- Cost for cutting beams
- Structure Weight
- Waste percentage (defined with the FFD Bin Packing algorithm)
- Steel Connection Cost

More information on how the geometry is evaluated can be found on these AU Classes:

- *ES119852 - Dynam(o)ite Your Steel Design @ AU Las Vegas 2017*
- *Optimization of Structural Designs with Dynamo @ AU London 2018*

At the end of this handout you'll find download links for these classes.

Datasets

The datasets used for the three presented cases in this class can be downloaded here:

<https://www.dropbox.com/s/m97mtmckvktunu8/BES224265%20Structural%20Dynam%28o%29ite%20Datasets.zip?dl=0>



Other references

More information on how to set up this workflow including the datasets, can be found from two other classes Dieter Vermeulen taught in the past:

ES9542: Dynam(o)ite Your Design for Engineers

In this class the principles of genetic algorithm optimization are explained on the optimization of a complex steel truss structure.

Class Recording:

<http://au.autodesk.com/au-online/classes-on-demand/class-catalog/classes/year-2015/robot-structural-analysis-professional/es9542#chapter=0>

ES119852 - Dynam(o)ite Your Steel Design @ AU Las Vegas 2017

In this class you could learn about how to involve computational design and optimization techniques in your structural steel constructions. The example of the "famous" Mars Torus Arena was used.

Class Recording:

<http://au.autodesk.com/au-online/classes-on-demand/class-catalog/classes/year-2017/dynamo-studio/es119852#chapter=0>

Presentation:

<https://www.dropbox.com/s/c5jq01n94tyr3jq/ES119852%20-%20Dynam%28o%29ite%20Your%20Steel%20Design.pptx?dl=0>

Datasets:

<https://www.dropbox.com/s/v33pyb27dhn0lj/ES119852%20-%20Dynam%28o%29ite%20Your%20Steel%20Design%20-%20datasets.zip?dl=0>

Optimization of Structural Designs with Dynamo @ AU London 2018

In this hands-on session, you could learn how to create your own Dynamo script for structural optimization with Optimo, Fractal and Refinery.

Presentation:

<https://www.dropbox.com/s/5jv73w47jgx6u87/Optimisation%20of%20Structural%20Designs%20with%20Dynamo.pptx?dl=0>

Datasets:

<https://www.dropbox.com/s/20l81yuw5hrquw9/Mars%20Torus%20Arena%20Datasets.zip?dl=0>

Intelligent optimization of steel structures in the early design phase through Visual Programming

Master's Thesis, July 2018, by Jonathan Rebien Olesen

<https://forum.dynamobim.com/t/using-genetic-algorithms-in-dynamo-for-generating-and-evaluating-design-alternatives-at-an-early-stage/25577>