Documentation – LFrame: Topology Optimization of Spatial Frames

Sarah Schroeder, Eduardo Lenz Cardoso

January, 2025

1 Introduction

If you are here, you were probably curious about the mathematical formulation of our project. This document will provide an overview of the theory.

First of all, let's assume that you are already familiar with the Finite Element Method and have a good understanding of the equilibrium problem, as well as the codes usually used to address this.

Thereby, this document will provide the foundations of Optimization and the subsequent process, including Topology Optimization (TO) and its formulation. We will begin with unconstrained optimization, move on to constrained optimization, and spend more time focusing on TO.

Any questions can be adressed to sarah.schroeder@udesc.edu.br, I'll be happy to help.

Note on Sources and References This document does not use formal references because the content is based on a academic analysis of methods and practices observed in the technical literature, field practices, and the development of the research. Although the ideas presented are not directly cited, they stem from concepts that are widely known and discussed in the field of topology optimization and structural analysis. It is a synthesis of personal experiences, applications, and improvements, with no intention of plagiarizing or presenting it as entirely novel.

2 Unconstrained Optimization

Unconstrained optimization aims to minimize x or $\mathbf{x}$ (scalar and vector forms) in a function *f(x)* by finding x, which, when applied to the function, gives its minimum value. To find x, a *descent direction* and a *step size* are needed, which *must* ensure at each iteration of the process that

$$f(\mathbf{x}^{(k+1)}) < f(\mathbf{x}^{(k)}), \tag{1}$$

where $\mathbf{x}^{k+1}$ is defined as:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)} \tag{2}$$

with: α = step size $\mathbf{d}^{(k)}$ = direction.
Thus, Eq. 1 can be rewritten as

$$f(\mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}) < f(\mathbf{x}^{(k)}), \tag{3}$$

where, using the Taylor series for linearizing the function, we arrive at the **descent condition**:

$$\vec{\nabla} f^T \mathbf{d} \leq 0. \tag{4}$$

This way, it is possible to work with the first algorithms for minimizing functions **without constraints**.

2.1 Steepest Descent

By deriving a function with respect to all its components (i.e., calculating the gradient), the resulting vector will indicate the direction of the function's growth. If the goal is to find its *minimum*, the opposite direction of the gradient should be considered. Hence, the direction is given by

$$\mathbf{d}^k = -\vec{\nabla} f(\mathbf{x}^k). \tag{5}$$

Additionally, a stopping criterion for the program can be

$$||\vec{\nabla} f||_2 < tolerance. \tag{6}$$

In other words, the gradient norm of the function must be smaller than the tolerance imposed in the program. Since the descent direction is determined by the gradient itself, a "small"direction vector means that we are very close to x^*, which is our objective.

Another important point is that the Hessian ($\nabla^2 f$) must be positive, ensuring that the critical point found is a local minimum. If the Hessian is positive definite, the function is concave upward at that point. Thus, by the definition of the Hessian matrix, it is also necessary to ensure that the function is twice differentiable.

Finally, the direction vector from Eq. 5 can be normalized as follows:

$$d = -\frac{\vec{\nabla} f}{||\vec{\nabla} f||_2}. \tag{7}$$

2.2 Line Search

After determining the descent direction, it is still necessary to find the ideal step size, α. It is important to note that, for now, only *unimodal* functions (those with a single minimum point, which is also the global minimum) will be considered. For non-unimodal functions, multiple minimum points (local minima) may exist. An example of a unimodal function is shown below:

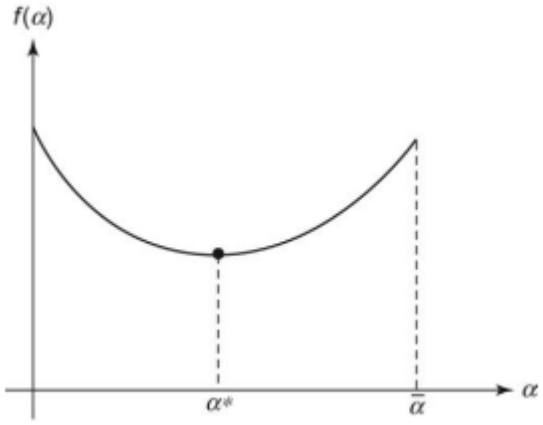

Figura 1: Unimodal function (ARORA)

The step size α can be determined using the Line Search method, which can be implemented through:

1. Random Search

As the name suggests, this method uses random numbers to evaluate the function at various points, storing the lowest value. An algorithm for this method could be as follows:

1. Define the number of iterations.
2. Calculate the function's value at a randomly generated point within a specified range.
3. Compare the result with the previous value. If it is smaller, the new result becomes the lowest value and the new comparison criterion.
4. Repeat the procedure until the defined number of iterations is reached.

2. Grid Search/Exhaustive Search

This method uses a fixed δ to scan the function, following the procedure below:

1. Evaluate the function at an initial point.
2. Add δ to the point ("move forward").
3. Evaluate the new point. If
$$f(x_0 + \delta) > f(x_0),$$
infer that the minimum lies between $f(x_0 + \delta)$ and $f(x_0)$, and refine δ using a predefined multiplication factor.
4. Stop refining δ after reaching a certain tolerance, and the minimum can be found as
$$x^* = \frac{f(x_0 + \delta) + f(x_0)}{2}.$$

3. Bracketing

This method divides the evaluated function range into four parts and refines these intervals, with the "outer"interval being $[a, b]$ and the inner one being $[c, d]$.

If $f(d) < f(c)$, the new interval is $[c, b]$ If $f(c) < f(d)$, the new interval is $[a, d]$
If $f(d) = f(c)$, the new interval is $[c, d]$

An important question is: *How is the division performed, and what defines the interval sizes?* The answer is not unique, and this method can be divided into two approaches: equal interval division or using the golden ratio to determine distances.

- Equal intervals: Here, the function is divided into equally spaced intervals, as shown in the figure below:

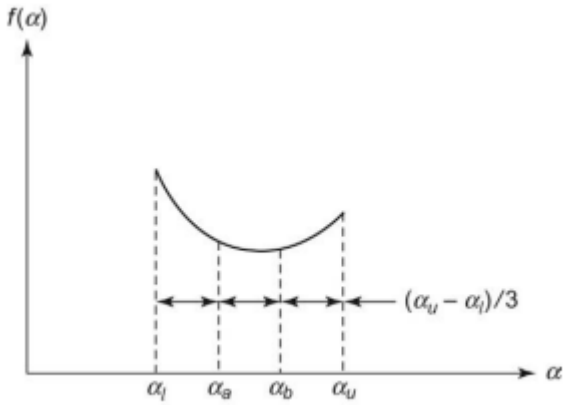

Figura 2: Bracketing with equal intervals

Thus, it is only necessary to define $[a, b]$.

- Using the Golden Ratio:
  Let two intervals be given as $[a, d] = A$ and $[d, b] = B$, such that the **total** interval relates to A in the same way A relates to B, and this is equal to ϕ:

$$\frac{A+B}{A} = \frac{A}{B} = \phi$$
$$\frac{\frac{A}{B}+1}{\frac{A}{B}} = \frac{A}{B}.$$

  Since

$$\frac{A}{B} = \phi,$$
$$\frac{\phi+1}{\phi} = \phi.$$

  This results in the quadratic expression:

$$\phi^2 - \phi - 1 = 0,$$

  which has one of its solutions as

$$\phi = \frac{1+\sqrt{5}}{2}, \tag{8}$$

  which is the **Golden Ratio**.

2.3 Newton's Method

The first method discussed is Newton's Method, which increases the "reach"of the descent direction by using the Taylor Series truncated at the second term, assuming that **the attraction point is already close**. Therefore, from the Taylor series, we have:

$$f(x^{(k+1)}) \approx f(x^{(k)}) + \vec{\nabla}^{(k)T} f^{(k)} + \frac{1}{2}\Delta x^{(k)} \mathbf{H}^{(k)} \Delta x^{(k)} \tag{9}$$

where $\mathbf{H}$ is the *Hessian Matrix* of the function:

$$H[f(x_1, x_2, ...x_n)] = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 x_2} \\ \frac{\partial^2 f}{\partial x_2 x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 x_1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n x_2} & \cdots & \frac{\partial^2 f}{\partial x^2 x_n} \end{pmatrix}. \tag{10}$$

Therefore, knowing that for $\mathbf{x}$ to be a minimum (or maximum), the derivative at the point must be *zero*, that is,

$$\vec{\nabla} f(x^{(k+1)}) = \mathbf{0}, \tag{11}$$

or

$$\frac{\partial f(x^{(k+1)})}{\partial \Delta x^{(k)}} = \mathbf{0}. \tag{12}$$

In Eq. 9, we have

$$\mathbf{0} + \vec{\nabla} f + \mathbf{H}^{(k)} \Delta x^{(k)} = \mathbf{0} \tag{13}$$

and, therefore:

$$\mathbf{H}^{(k)} \Delta x^{(k)} = -\vec{\nabla} f^{(k)}. \tag{14}$$

Thus, the equation above can be treated as a system of equations where $\Delta x^{(k)}$ is the unknown. Therefore,

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mathbf{H}^{(k)-1} \vec{\nabla} f^{(k)} \tag{15}$$

becomes

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \Delta x. \tag{16}$$

The fact that the series is expanded to the second derivative (from which the Hessian matrix arises) increases the possibilities for the search direction, making Newton's Method much more efficient than Steepest Descent. However, there are some issues with this method. It is still necessary to define the step α, take the first step, and ensure that $\mathbf{H}$ is positive definite.
For the step:
In the case of Newton's Method without modification, one can use

$$\alpha = 1.0 \tag{17}$$

How the first step will be taken:
Steepest Descent will be used for the initial step. In the code, this can be understood as changing the first iteration to

$$I \Delta x^{(k)} = -\vec{\nabla} f^{(k)} \tag{18}$$

To ensure that the Hessian is positive definite, something similar to the last method that will be discussed will be used.

2.4 Modified Newton's Method

For this modification, α will no longer be zero, and instead, it will come from the usual *Line Search*.

2.5 Levenberg-Marquardt Method

This method modifies the Hessian to ensure that the matrix is positive definite. For this, *all eigenvalues of* $\boldsymbol{H}$ *must be greater than zero*. Thus, the eigenvalues of the Hessian can be calculated, and if one of them is less than zero, $\mathbf{H}$ is modified to

$$\mathbf{H} = \mathbf{H} + \beta|\bar{\lambda}|I. \tag{19}$$

Where:
β = multiplication factor
$\bar{\lambda}$ = smallest eigenvalue
I = identity matrix

In the Levenberg-Marquardt method, β is a quite high value. Here, smaller values (such as 1.5, for example) will be used, just to ensure that the eigenvalues are guaranteed to be positive. A question that might arise is whether we can actually do this without compromising the results. The answer is quite simple: Even though the Hessian is multiplied, the information regarding the *curvature* of the function ("the variation of the variation of the function") remains intact. And this is precisely the information that matters in this case. Therefore, this can be done without problems.

2.6 Conjugate Gradient Method

Also called the *Deflection Method*, this procedure eliminates the need to compute the Hessian matrix by hiding it in the calculation of the descent direction. In higher-dimensional problems, this represents significant computational time savings.
This method uses the concept of orthogonality in an amplified form. We know that, given $\mathbf{u}$ and $\mathbf{v}$ as two vectors, orthogonality is guaranteed if

$$\mathbf{u} \cdot \mathbf{v} = 0.$$

This is the same as

$$\mathbf{u}^T I \mathbf{v} = 0,$$

where I is the identity matrix. Thus, by expanding this concept, we can say that $\mathbf{u}$ and $\mathbf{v}$ are *A-conjugates* if

$$\mathbf{u}^T \mathbf{A} \mathbf{v} = 0, \tag{20}$$

where $\mathbf{A}$ is any matrix that is *positive definite and symmetric*. Applying this concept to two consecutive descent directions of the function being optimized, we have

$$\mathbf{d}^{(k-1)T} \mathbf{A} \mathbf{d}^{(k)} = 0. \tag{21}$$

Approximating the function by the Taylor Series truncated at the second term

$$f(\mathbf{x}) \approx c + \mathbf{b}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} \tag{22}$$

and, for the present method,

$$\mathbf{d}^k = -\nabla_x f(\mathbf{x}^k) + \beta^k \mathbf{d}^{k-1} \tag{23}$$

where β^k is a value such that Eq. 20 is satisfied. Substituting Eq. 23 into Eq. 20, we get

$$\mathbf{d}^{(k-1)T}\mathbf{A}(-\nabla_x f(\mathbf{x}^k) + \beta^k \mathbf{d}^{k-1}) = 0 \tag{24}$$

2.7 Side Constraints

So far, we have worked with problems of the form

$$\begin{cases} x^{(k+1)} = x^{(k)} + \alpha^{(k)}\mathbf{d}^{(k)} \\ \mathbf{d}^{(k)}\nabla f^{(k)} < 0 \\ \alpha^{(k)} = \arg\min f(x^{(k)} + \alpha\mathbf{d}^{(k)}) \end{cases}$$

where

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)}\mathbf{d}^{(k)} \rightarrow \text{point update}$$

$$\mathbf{d}^{(k)}\nabla f^{(k)} < 0 \rightarrow \text{minimization condition}$$

$$\alpha^{(k)} = \arg\min f(x^{(k)} + \alpha\mathbf{d}^{(k)}) \rightarrow \textit{Line Search} \text{ principle.}$$

With side constraints, the problems are viewed as

$$\begin{cases} \min f(\mathbf{x}) \\ \mathbf{x} \in S \subseteq \mathbb{R}^n \\ \mathbf{x}_l \leq \mathbf{x} \leq \mathbf{x}_u \end{cases}$$

With

$$\min f(\mathbf{x}) \rightarrow \text{minimization of } f(\mathbf{x})$$

$\mathbf{x} \in S \subseteq \mathbb{R}^n \rightarrow$ where $\mathbf{x}$ belongs to S which is contained within the set of real numbers of dimension n

$\mathbf{x}_l \leq \mathbf{x} \leq \mathbf{x}_u \rightarrow$ and is bounded between the lower bound ($\mathbf{x}_l$) and the upper bound ($\mathbf{x}_u$)

Visually, this can be understood as follows:

Let a function be given by

$$f(\mathbf{x}) = (x_1 - 4)^2 + (x_2 - 4)^2 \tag{25}$$

With the minimum point at

$$\mathbf{x}^* = (4, 4)$$

Graphically, in two dimensions, it would look like Fig. 2.7:

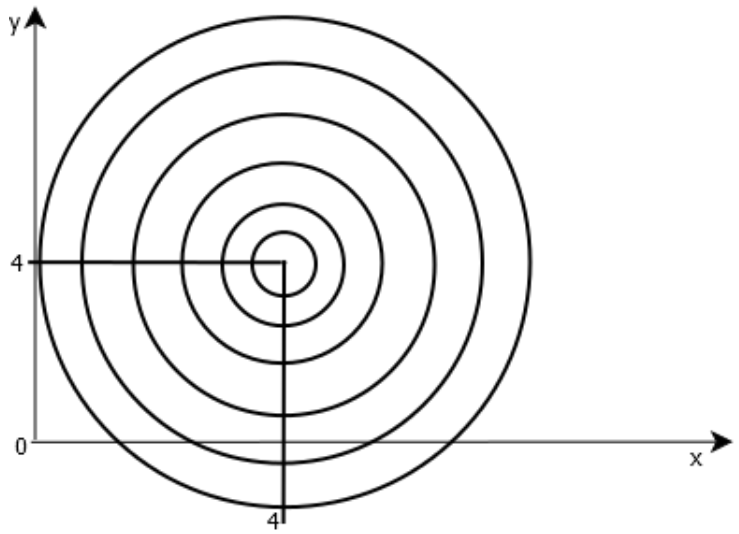

Figura 3: Function with explicit minimum point

Let

$$x_{1l} = 2$$
$$x_{1u} = 6$$
$$x_{2l} = 5$$
$$x_{2u} = +\infty$$

3 Constrained Optimization - Equality and Inequality Constraints

3.1 Equality Constraints

Let the optimization problem be of the type

$$\begin{cases} min(x_1 - 3)^2 + (x_2 - 4)^2 \\ s.t. \\ x_1 + x_2 = 8 \end{cases}, \tag{26}$$

the last line represents an *equality constraint*. Graphically, it would look like the figure below.

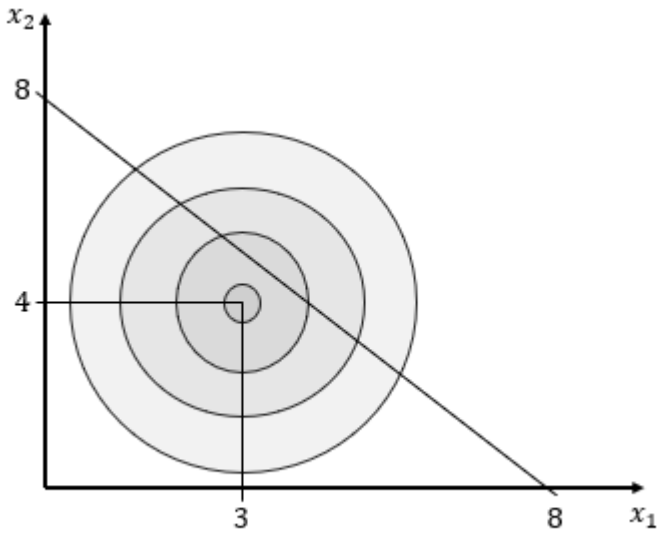

Figura 4: Graphical representation of an equality constraint.

The solution to the problem must be *exactly* on the line formed by $(8, 8)$**.**

If there were no constraints, it is known that the minimum point of a function can be found using the concept that the derivative at a maximum/minimum point of a function is zero, i.e.,

$$\frac{df}{dx} = \nabla f = \mathbf{0}. \tag{27}$$

However, in the case of equality constraints, this is not suitable, because, as seen, the solution is restricted by some condition imposed. Therefore, for this type of problem, it is appropriate to introduce *a new variable for each equality constraint*. This new objective function will be called the *Lagrangian Function* or *Composite Objective Function*. The new variables will be introduced here as

λ. Using the problem from the beginning of the section, its Lagrangian function is

$$\mathcal{L}(x_1, x_2, \lambda) = (x_1 - 3)^2 + (x_2 - 4)^2 + \lambda(x_1 + x_2 - 8). \tag{28}$$

Note that the term accompanying λ is exactly the equality constraint of the problem:

$$x_1 + x_2 = 8 \rightarrow x_1 + x_2 - 8 = 0.$$

Furthermore, using the same idea from Eq. 61 for unconstrained problems, the derivatives of $\mathcal{L}$ with respect to x_1, x_2, and λ can be obtained.

$$\frac{d\mathcal{L}}{dx_1} = 2(x_1 - 3) + \lambda = 0 \rightarrow x_1 = \frac{-\lambda + 6}{8} \tag{29}$$

$$\frac{d\mathcal{L}}{dx_2} = 2(x_2 - 4) + \lambda = 0 \rightarrow x_2 = \frac{-\lambda + 8}{2} \tag{30}$$

$$\frac{d\mathcal{L}}{d\lambda} = x_1 + x_2 - 8 = 0 \rightarrow \left(\frac{-\lambda + 6}{3}\right) + \left(\frac{-\lambda + 8}{2}\right) = 8 \tag{31}$$

Solving Eq. 65 gives

$$\lambda^* = -1, \;\; \therefore \; x_1^* = 3.5; \; x_2^* = 4.5. \tag{32}$$

Now, writing a general optimization problem with an equality constraint, we have

$$\begin{cases} min \; f(x) \\ s.t. \\ h_j(\mathbf{x}) = 0; j = 1, 2, \ldots m_h \end{cases}, \tag{33}$$

and the Lagrangian function can be written as

$$\mathcal{L}(\mathbf{x}, \lambda) = f(\mathbf{x}) + \sum_{j=1}^{m_h} \lambda_j h_j(\mathbf{x}), \tag{34}$$

where the vector λ contains the *Lagrange multipliers* and m_h is the number of equality constraints in the problem. Using the conditions derived from the derivative concept as shown in the example,

$$\begin{cases} \nabla_x \mathcal{L} = \mathbf{0}|_{m \times 1} \\ \nabla_\lambda \mathcal{L} = \mathbf{0}|_{m_h \times 1} \end{cases}. \tag{35}$$

The first line of Eq. 69 can be developed to give

$$\nabla_x \mathcal{L} = \nabla_x f(\mathbf{x}) + \sum_{j=1}^{m_h} \lambda_j \nabla h_j(\mathbf{x}) = \mathbf{0} \tag{36}$$

and, therefore,

$$\nabla f(\mathbf{x}^*) = -\sum_{j=1}^{m_h} \lambda_j^* \nabla_x h_j(\mathbf{x}^*). \tag{37}$$

An important point to highlight is that $\mathbf{x}^*$ and λ^* are called "regular optima"if $\nabla_x h_j$ are linearly independent.

3.2 Inequality Constraints

It is also possible (and more common) for there to be problems of the type

$$\begin{cases} min\ f(x) \\ s.t. \\ g_j(\mathbf{x}) \leq 0\ j = 1 \dots m_g \end{cases}, \tag{38}$$

where the feasible solution space looks something like

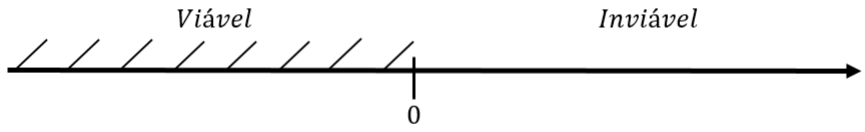

Figura 5: Graphical representation of an inequality constraint

Thus, there are now feasible and infeasible intervals for each constraint. It can be observed that Eq. 34 is no longer suitable in these cases due to the inequality conditions. The appropriate approach is to introduce another set of variables, called here *slack* variables. The Lagrangian for problems with inequality constraints becomes

$$\mathcal{L}(\mathbf{x}, \mu, \mathbf{s}) = f(\mathbf{x}) + \sum_{j=1}^{m_g} \mu_j \underbrace{(g_j(\mathbf{x}) + s_j^2)}_{h}. \tag{39}$$

The above equation, developed by Karush-Kuhn-Tucker (KKT), ensures that the results obtained are adjusted by s^2 if they lie within the feasible solution interval and, if the result is in an infeasible solution interval, there is no point in making this adjustment. Therefore, this variable takes a value of zero while the remaining term inside the summation increases, indicating the importance of the constraint that made a particular interval infeasible.

Thus, there are now three gradients and conditions that need to be considered:

$$\left\{ \nabla_x h = \mathbf{0} | n \times 1 \quad \nabla \mu h = \mathbf{0} | mg \times 1 \; \nabla_s h = \mathbf{0} | mg \times 1 \quad \right. , \tag{40}$$

In the previous example, applying a constraint of inequality...

$$\left\{ min \; (x_1 - 3)^2 + (x_2 - 4)^2 \quad tq \; x_1 + x_2 \geq 8 \rightarrow -x_1 - x_2 + 8 \leq 0 = g_1(\mathbf{x}) \quad \right. , \tag{41}$$

Important: Note that the inequality condition has been modified. This should be done whenever the condition is "greater than or equal,"as the method was developed for "less than or equal"conditions.

The Lagrangian function for this current example is

$$\mathcal{L} = (x_1, x_2, \mu_1, s_1) = (x_1 - 3)^2 + (x_2 - 4)^2 + \mu_1(-x_1 - x_2 + 8 + s_1^2) \tag{42}$$

From Eq. 74,

$$\frac{d\mathcal{L}}{dx_1} = 2(x_1 - 3) - \mu_1 = 0 \tag{43}$$

$$\frac{d\mathcal{L}}{dx_2} = 2(x_2 - 4) - \mu_1 = 0 \tag{44}$$

$$\frac{d\mathcal{L}}{\mu_1} = -x_1 - x_2 + 8 + s^2 = 0 \tag{45}$$

$$\frac{d\mathcal{L}}{s_1} = 2\mu_1 s = 0 \tag{46}$$

Where Eq. 46 is called the *complementary slackness condition*. Notice that for this equation to hold, either $\mu_1 = 0$ or $s_1 = 0$. In the first case, this means that the interval is infeasible, and the *slack* variable is making the correction, with g_1 being called the ***inactive condition***. In the second case, the interval is already feasible, and s is unnecessary, with all the weight going to μ_1, and g_1 is called the ***active condition***. An artificial way to use the slackness condition is to use the condition

$$\{\mu_j g_j = 0 \quad . \tag{47}$$

4 Introduction to Topology Optimization

Topology Optimization, unlike Parametric Optimization, does not require a pre-existing design to modify parameters. A pre-defined continuous space, "filled"with the material that makes up the structure, is used. Therefore, the goal of Topology Optimization is to remove material spaces in the best possible way while adhering to the imposed constraints.

A continuous structure can be understood as many parts of specific elements. Here, 3D frame elements will be used to form structures like the one below:

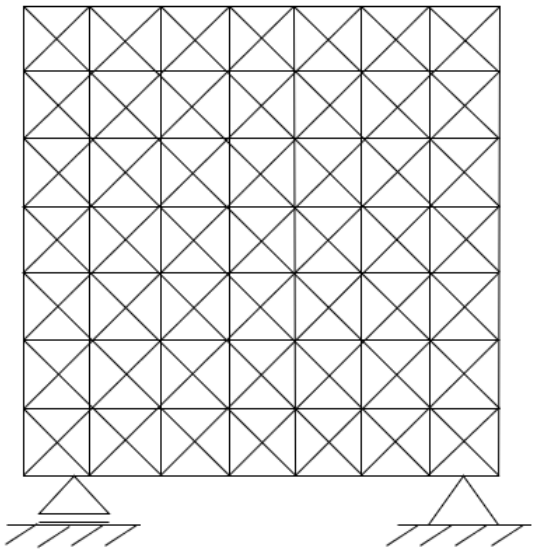

Figura 6: Discretization of a continuous space into 3D frame elements

The objective here is to remove as many frame elements as possible while adhering to a specific constraint, such as minimal mass. Hence, we can understand this problem as having a volume constraint, where the goal is to achieve

the maximum stiffness for the structure. Therefore, a topology optimization problem can be understood as

$$\begin{cases} max\ stiffness \\ S.t \\ V \leq \bar{V} \end{cases} \tag{48}$$

Thus, knowing that the external work acting on the structure is

$$\tau_{ext} = \frac{1}{2} F u, \tag{49}$$

where

$$F = \text{applied force}$$
$$u = \text{displacement},$$

It is also known that

$$F = KU \rightarrow U = \frac{F}{K},\ \therefore \tau_{ext} = \frac{1}{2}\frac{F^2}{K}. \tag{50}$$

Where K represents stiffness. Notice that increasing stiffness reduces the external work. That is, the objective can be understood as minimizing the external work applied to the structure while maintaining a volume constraint (for example).

$$\begin{cases} min\ \tau_{ext} \\ tq \\ V \leq \bar{V} \end{cases} \tag{51}$$

Additionally, $\tau_{ext} = Fu$ can be understood as "Compliance", C, or *flexibility*. Therefore, the objective function will be

$$\mathbf{C} = \mathbf{F}^T \mathbf{U}. \tag{52}$$

Thus, following the theory already discussed, a new variable needs to be defined, which in this case will be the relative density of each element. This variable will be defined as belonging to an interval between (not including) 0 (where theoretically there will be no material) and 1 (where the element remains positioned):

$$\rho_e \in (0, 1) \tag{53}$$

and the longitudinal elastic modulus of each element becomes dependent on this relative density:

$$E_e = \rho_e E_e^0 \tag{54}$$

Similarly, the volume of the element will be

$$V_e = \rho_e V_e^0 = \rho_e L_e A_e \tag{55}$$

and finally, stiffness is defined as

$$\mathbb{K}_e^{local} = \rho_e \mathbb{K}^{0local}. \tag{56}$$

In the full structure, the volume and stiffness dependent on ρ will be the summations across all elements, i.e.,

$$V(\vec{\rho}) = \sum_{e=1}^{n} V_e = \sum_{e=1}^{n} \rho_e L_e A_e, \tag{57}$$

$$\mathbb{K}(\vec{\rho}) = \sum_{e=1}^{n} \mathbb{H}_e^T \mathbb{R}_e^T \mathbb{K}_e^{0local} \mathbb{R}_e \mathbb{H}_e = \sum_{e=1}^{n} \mathbb{H}_e^T \mathbb{R}_e^T \rho_e \mathbb{K}_e^{local} \mathbb{R}_e \mathbb{H}_e, \tag{58}$$

where $\mathbb{H}$ is a location matrix and $\mathbb{R}$ is the rotation matrix of the element, deduced in the previous material. If $\mathbf{F}$ is given,

$$\mathbb{K}(\vec{\rho})\mathbf{U}(\vec{\rho}) = \mathbf{F} \tag{59}$$

Rewriting the optimization problem, we have

$$\begin{cases} min\ \mathbf{F}^T \mathbf{U}(\vec{\rho}) \\ tq \\ \mathbb{K}(\vec{\rho})\mathbf{U}(\vec{\rho}) = \mathbf{F} \\ V \leq \bar{V} \end{cases} \tag{60}$$

The Lagrangian becomes

$$\mathcal{L}(\vec{\rho}, \mu) = \underbrace{\mathbf{F}^T \mathbf{U}(\vec{\rho})}_{f(\vec{\rho})} + \mu(V(\vec{\rho}) - \bar{V}), \tag{61}$$

where $f(\vec{\rho})$ is the flexibility function. The derivatives are

$$\frac{d\mathcal{L}}{d\vec{\rho}} = \frac{df(\vec{\rho})}{d\rho} + \mu \frac{dV(\rho)}{d\rho} \tag{62}$$

and

$$\frac{d\mathcal{L}}{d\vec{\rho}} = V(\rho) - \bar{V} = 0. \tag{63}$$

Manipulating Eq. 96, considering that it will be used for all elements and dividing the whole equation by $\mu \frac{dV(\rho)}{d\rho_k}$, we get

$$\frac{df(\vec{\rho})}{d\rho_k} / \mu \frac{dV(\vec{\rho})}{d\rho_k} = -1 \ , \ k = 1...n \tag{64}$$

Letting

$$\frac{df(\vec{\rho})}{d\rho_k} / \mu \frac{dV(\vec{\rho})}{d\rho_k} = \beta_k, \tag{65}$$

it is known that $\beta_k = -1 \ \forall \ k$.
Thus, we have the optimal criteria,

$$\frac{df}{d\rho_k}, \frac{dV}{d\rho_k} \ and \ \mu. \tag{66}$$

Derivative calculations:
Starting with

$$f(\rho) = \mathbf{F}^T \mathbf{U}(\vec{\rho}). \tag{67}$$

Differentiating both sides and using the chain rule:

$$\frac{df}{d\rho_k} = \underbrace{\frac{d\mathbf{F}^T}{d\rho_k}\mathbf{U}(\vec{\rho})}_{0} + \mathbf{F}^T \frac{d\mathbf{U}(\rho)}{d\rho_k} \tag{68}$$

$$\tag{69}$$

thus

$$\frac{df}{d\rho_k} = \mathbf{F}^T \frac{d\mathbf{U}(\rho)}{d\rho_k}. \tag{70}$$

Since

$$\mathbb{K}(\vec{\rho})\mathbf{U}(\vec{\rho}) = \mathbf{F}, \tag{71}$$

we can differentiate this expression:

$$\frac{d\mathbb{K}}{d\rho_k}\mathbf{U}(\vec{\rho}) + \mathbb{K}\frac{d\mathbf{U}(\vec{\rho})}{d\rho_k} = \mathbf{0} \rightarrow \mathbb{K}\frac{d\mathbf{U}(\vec{\rho})}{d\rho_k} = -\frac{d\mathbb{K}}{d\rho_k}. \tag{72}$$

Multiplying both sides by $\mathbb{K}^{-1}$,

$$\underbrace{\mathbb{K}^{-1}\mathbb{K}}_{\mathbb{I}}\frac{d\mathbf{U}(\vec{\rho})}{d\rho_k} = -\mathbb{K}^{-1}\frac{d\mathbb{K}}{d\rho_k}\mathbf{U}(\vec{\rho}) \rightarrow \frac{d\mathbf{U}(\vec{\rho})}{d\rho_k} = -\mathbb{K}^{-1}\frac{d\mathbb{K}}{d\rho_k}\mathbf{U}(\vec{\rho}). \tag{73}$$

Substituting the expression above into Eq. 70,

$$\frac{df}{d\rho_k} = -\mathbf{F}^T\mathbb{K}^{-1}(\vec{\rho})\frac{d\mathbb{K}}{d\rho_k}\mathbf{U}(\vec{\rho}). \tag{74}$$

Since $\mathbb{K}^T = \mathbb{K}$ and $\mathbb{K}\mathbf{U} = \mathbf{F}$,

$$\boxed{\frac{df}{d\rho_k} = -\mathbf{U}^T\frac{d\mathbb{K}}{d\rho_k}\mathbf{U}(\vec{\rho}).} \tag{75}$$

Thus, the variation in ρ_k perturbs $f(\rho)$ (the compliance function), which in turn perturbs $\mathbb{K}_{global}$ and influences $\mathbf{U}_{global}$!

However, in the above equation, there is still the need to solve the derivative of the stiffness matrix. It is known that

$$\mathbb{K}(\vec{\rho}) = \sum_{e=1}^{n}\mathbb{H}_e^T\mathbb{R}_e^T\rho_e\mathbb{K}_e^{0,local}\mathbb{R}_e\mathbb{H}_e, \tag{76}$$

where $\mathbb{H}$ is a location matrix of dimension $12\times$ total number of degrees of freedom.

Taking the derivative on both sides with respect to the relative density of the interaction,

$$\frac{d\mathbb{K}(\vec{\rho})}{d\rho_k} = \sum_{e=1}^{n}\mathbb{H}_e^T\mathbb{R}_e^T\frac{d\rho_e}{d\rho_k}\mathbb{K}_e^{0,local}\mathbb{R}_e\mathbb{H}_e. \tag{77}$$

To assist in solving this equation, the Kronecker delta function is used, which states that

$$\delta_{ek} = \begin{cases} 1 & \text{if } e = k \\ 0 & \text{if } e \neq k \end{cases}, \tag{78}$$

that is, if the interaction is within the element in question, the function is 1, but if it is in any other element, it is 0. This eliminates practically all the terms in the summation if we make

$$\mathbb{K}(\vec{\rho}) = \sum_{e=1}^{n}\mathbb{H}_e^T\mathbb{R}_e^T\delta_{ek}\mathbb{K}_e^{0,local}\mathbb{R}_e\mathbb{H}_e, \tag{79}$$

which simplifies to

$$\boxed{\mathbb{K}(\vec{\rho}) = \mathbb{H}_k^T \mathbb{R}_k^T \mathbb{K}_k^{0,local} \mathbb{R}_k \mathbb{H}_k} \tag{80}$$

and therefore,

$$\frac{df}{d\rho_k} = -\mathbf{U}^T \frac{d\mathbb{K}}{d\rho_k} \mathbf{U}(\vec{\rho}). \tag{81}$$

Substituting this into Eq. 75,

$$\boxed{\frac{df}{d\rho_k} = -(\mathbf{U}^T \mathbb{H}_k^T) \mathbb{R}_k^T \mathbb{K}_k^{0,local} \mathbb{R}_k (\mathbb{H}_k \mathbf{U}(\vec{\rho}))}. \tag{82}$$

Since

$$\mathbf{U}^T \mathbb{H}_k^T = \mathbf{u}_k^T \text{ (displacement vector of the element), and}$$
$$\mathbb{R}_k^T \mathbb{K}_k^{0,local} \mathbb{R}_k = \mathbb{K}_k^{0,global},$$

df can be written as

$$\frac{df}{d\rho_k} = -\mathbf{u}_k^T \mathbb{K}_k^{0,global} \mathbf{u}_k. \tag{83}$$

For the derivative of the volume,

$$\frac{dV(\rho)}{d\rho_k} = \sum \frac{d\rho_e}{d\rho_k} L_e A_e = L_k A_k \tag{84}$$

4.1 Augmented Lagrangian - Pure Penalization

An alternative form of the Lagrange function is

$$L(\mathbf{x}) = f(\mathbf{x}) + \sum_{j=1}^{h} r_j g_j(\mathbf{x})^2, \tag{85}$$

where r is a penalty term presented as a constant. As r increases, the minimum of the function tends toward the constraint, that is,

$$\lim_{r \to \infty} = value\ of\ the\ constraint. \tag{86}$$

However, a problem with this method is that, numerically, if $r \to \infty$, only the constraints would be "seen". Therefore, it is necessary to circumvent this issue.

4.2 Augmented Lagrangian - External Penalization

In this case, Eq. 85 becomes

$$L(\mathbf{x})^k = f(\mathbf{x}) + \frac{r^k}{2}\sum_{j=1}^{m}\left\langle \frac{\mu_j^k}{r^k}\mathbf{g}_j(\mathbf{x})\right\rangle^2. \tag{87}$$

Where k is the *external iteration*. Thus, the objective of external penalization is to apply "shifts"to the constraints via the fine-tuning term $\frac{\mu_j^k}{r^k}$ *while* maintaining the penalization. The function inside the summation, squared, is understood as a multiplication by the Heaviside function, which zeros out negative terms — that is, if the constraint is inactive, the augmented Lagrangian function becomes the function $f(\mathbf{x})$ itself.

To satisfy the KKT conditions,

$$\nabla L = 0 = \nabla f + \sum \mu_j^k \nabla g(x^k) = 0 \tag{88}$$

and

$$\nabla_\mu L = g_j \mu_j = 0. \tag{89}$$

An important point to highlight is the definition of the first penalty term, r^0, given by

$$f(\mathbf{x}^0) + \frac{r^0}{2}\sum_{j=1}^{m}\left\langle g_j(\mathbf{x}^0)\right\rangle^2 \approx d, \tag{90}$$

where d is a real constant. Isolating the term of interest,

$$r^0 \approx \frac{2(d - f(\mathbf{x}^0))}{\sum_{j=1}^{m}\left\langle g_j(\mathbf{x}^0)\right\rangle^2} \tag{91}$$

or

$$r^0 \approx \frac{2f(\mathbf{x}^0)}{\sum_{j=1}^{m}\left\langle g_j(\mathbf{x}^0)\right\rangle^2}. \tag{92}$$

Thus, the optimization problem reduces to

$$P^k = \{min\ L^k(\mathbf{x}); k = 1, ..., n_k. \tag{93}$$

Where, now, several optimization *loops* are performed to reach the result, which seeks to minimize the augmented Lagrangian function.

5 Structural Optimization

In this way, an entire structure can be optimized, given the imposed loads and supports used. The implemented code follows the logic below:

1. The *.yaml* input file is generated based on the desired distribution of the structure's elements and supports;
2. The file is read, and the mesh is generated in the *gmsh* format;
3. The total volume of the structure and the volume constraint must be specified;
4. The optimization loops begin, where:
   (a) The relative density vector is calculated;
   (b) The derivatives of compliance and volume are computed;
   (c) The stiffness matrix, rotation, etc., are calculated (standard procedure);
   (d) The external penalization method is applied. If the constraint is not met, the optimization loop repeats. Otherwise, it exits.
5. A results mesh is generated, showcasing the optimized structure.

6 Inclusion of Additional Constraints - Displacement

6.1 The Constraint

Consider any body subjected to a specific load that causes displacement in $\hat{j}$. This displacement can be constrained to a certain value.

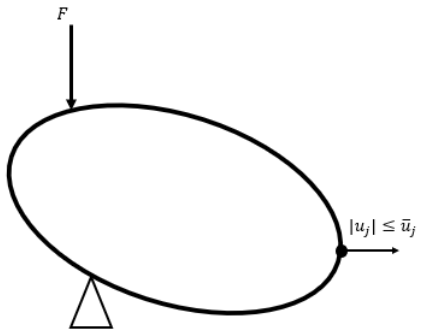

Figura 7: Representation of a constrained displacement

Thus, the constraint can be expressed as

$$g(\rho) = u_j(\rho)^2 \leq \bar{u}_j^2. \tag{94}$$

However, different constraints can have different orders of magnitude, necessitating normalization. Starting from the constraint presented above, dividing all terms by $\bar{u}_j^2$, the normalized constraint becomes

$$\frac{u_j(\boldsymbol{\rho})}{\bar{u}_j^2} \leq 1 \rightarrow \frac{u_j(\boldsymbol{\rho})}{\bar{u}_j^2} - 1 \leq 0, \tag{95}$$

and this normalized constraint will be used in the augmented Lagrange function. Similarly, we need the derivative of the constraint:

$$\frac{dg}{d\boldsymbol{\rho}_m} = \frac{d}{d\boldsymbol{\rho}_m}\left(\frac{u_j(\boldsymbol{\rho})}{\bar{u}_j^2} - 1\right) \rightarrow \frac{dg}{d\boldsymbol{\rho}_m} = 2\frac{1}{\bar{u}_j^2}u_j(\boldsymbol{\rho})\frac{du_j(\boldsymbol{\rho})}{d\rho_m}. \tag{96}$$

However, a previously encountered problem is deriving the displacement with respect to ρ_m. To address this, we use the location vector:

$$\mathbf{L}_j = \begin{pmatrix} 0 \\ 0 \\ \cdots \\ j \\ 0 \\ \cdots \end{pmatrix}, \tag{97}$$

which will "extract"the j location such that

$$u_j = \mathbf{L}_j^T \mathbf{U}. \tag{98}$$

Thus, we have:

$$\frac{dg}{d\rho_m} = 2\frac{1}{\bar{u}_j^2}u_j(\boldsymbol{\rho})\mathbf{L}_j^T\frac{d\mathbf{U}(\boldsymbol{\rho})}{d\rho_m}. \tag{99}$$

Knowing that

$$\mathbb{K}\mathbf{U} = \mathbf{F}, \tag{100}$$

$$\frac{d\mathbb{K}(\boldsymbol{\rho})}{d\rho_m}\mathbf{U} + \mathbb{K}(\boldsymbol{\rho})\frac{d\mathbf{U}(\boldsymbol{\rho})}{d\rho_m} = \mathbf{0}. \tag{101}$$

This leads to:

$$\frac{dg}{d\rho_m} = \frac{-2}{\bar{u}_j^2} u_j(\boldsymbol{\rho}) \mathbf{L}_j^T \mathbb{K}(\boldsymbol{\rho})^{-1} \frac{d\mathbb{K}(\boldsymbol{\rho})}{d\rho_m} \mathbf{U}(\boldsymbol{\rho}). \tag{102}$$

This computation is expensive due to the calculation of $\frac{d\mathbb{K}(\boldsymbol{\rho})}{d\rho_m}$. Thus, an alternative approach is necessary.

6.2 Adjoint Problem

The derivation using the adjoint problem bypasses the issue previously encountered by introducing a new adaptation in the Lagrange function. To understand the need for the adjoint problem, let us examine what would happen if we used the Augmented Lagrange Function,

$$\frac{dL_A}{d\rho_m} = \frac{df(\boldsymbol{\rho})}{d\rho_m} + \frac{d}{d\rho_m} \left(\frac{c^k}{2} \sum_{j=1}^{m} \left\langle \frac{\mu_j^k}{c^k} + \mathbf{g}_j \right\rangle^2 \right) \tag{103}$$

which results in

$$\frac{dL_A}{d\rho_m} = \frac{df(\boldsymbol{\rho})}{d\rho_m} + \frac{d}{d\rho_m} c^k \sum_{j=1}^{m} \left\langle \frac{\mu_j^k}{c^k} + \mathbf{g}_j \right\rangle \underbrace{\frac{dg_j}{d\rho_m}}_{\text{not worth it!}} . \tag{104}$$

Using the adjoint problem, we add a "neutral term,"the "adjoint vector,"denoted by $\boldsymbol{\lambda}$:

$$f + \frac{c}{2} \sum_{j=1}^{m} \left\langle \frac{\mu_j^k}{c^k} + \mathbf{g}_j \right\rangle^2 + \boldsymbol{\lambda}^T \underbrace{\mathbb{K}\mathbf{U} - \mathbf{F}}_{\mathbf{0}}. \tag{105}$$

Thus, the adjoint problem becomes:

$$\mathbb{K}\boldsymbol{\lambda} = -c^k \sum_{j=1}^{m} \left\langle \frac{\mu_j^k}{c^k} + \mathbf{g}_j \right\rangle \frac{\partial g_j}{\partial \mathbf{U}} \tag{106}$$

Here, $\frac{\partial g_j}{\partial \mathbf{U}}$, representing the variation of the constraints with respect to the displacement vector, can be calculated as:

$$\frac{\partial g_j}{\partial \mathbf{U}} = \frac{2}{\bar{u}_j^2} u_j \mathbf{L}_j \tag{107}$$

The implementation of this can be found in the supplementary document, which contains all the mentioned codes.

7 Derivative Verification

To verify the results of a numerically computed derivative, we can use the central finite difference method. Let us apply this approach to the derivative of the Augmented Lagrange function. According to the Taylor Series, truncated at the second term, we have:

$$f(x+\Delta x) \approx f(x) + \frac{df}{dx}\Delta x + \frac{1}{2}\frac{d^2 f(x)}{dx^2}\Delta x^2 + \ldots \rightarrow \frac{df(x)}{dx} \approx \frac{f(x+\Delta x) - f(x)}{\Delta x} \quad (108)$$

Thus, applying the concept above to dL_A:

$$\frac{dL_A(\boldsymbol{\rho})}{d\rho_m} \approx \frac{L_A(\boldsymbol{\rho}+\delta) - L_A(\boldsymbol{\rho})}{|\delta|}, \quad (109)$$

where δ is a perturbation equivalent to Δx in the usual notation.

An important observation is that care must be taken when arbitrating the value of δ, as a large value results in truncation errors (stemming from the Taylor Series used to derive the expression), and a very small value causes rounding errors. This is especially critical when implementing this concept in a computational code, as a very small δ may exceed the machine's ϵ, leading to a false zero result.

8 Stress Constraints

8.1 Definition

It is known that at a critical point of a beam element, the equivalent stress must be less than a prescribed stress value. Computationally, this can be expressed as:

$$g_i = \sigma_j \leq \bar{\sigma}_j, \quad (110)$$

where σ_j can be written as a function depending on the axial and shear stresses at the point of analysis:

$$f(\sigma) = \sqrt{\sigma_{xx}^2 + A\sigma_{xy}^2}, \quad (111)$$

where A takes the value of 3 for the von Mises equivalent stress criterion or 4 for the Tresca criterion. It is important to note that the material is considered ductile and that the stresses in the y direction are null, i.e.,

$$\sigma_{yy} = 0. \quad (112)$$

Thus, from Eq. 110, we have:

$$g_j = \frac{\sigma_{eqj}}{\bar{\sigma}_j} - 1 \leq 0. \tag{113}$$

9 Internal Forces

Considering a beam element as depicted in Fig. 8, the internal forces at node 1 can be represented as:

$$N(F_4, x) \tag{114}$$
$$V_y(F_3, x, q_y) \tag{115}$$
$$V_z(F_2, x, q_z) \tag{116}$$
$$T(F_4, x) \tag{117}$$
$$M_z(F_3, q_y, F_5, x) \tag{118}$$
$$M_y(F_2, q_z, F_6, x). \tag{119}$$

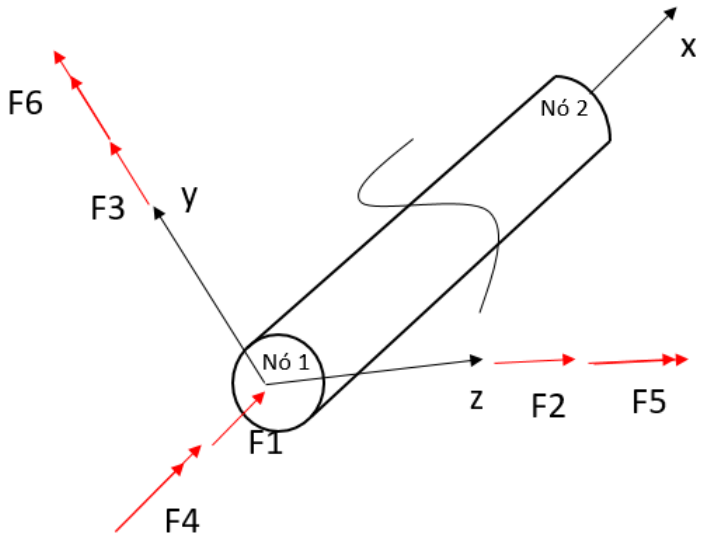

Figura 8: 3D beam element

Considering the long beam theory, we know:

$$\sigma_{xx} = f_1(N, M_y, M_z) \tag{120}$$
$$\sigma_{xy} = f_2(T), \tag{121}$$

then Eq. 111 becomes:

$$f(\sigma) = \sqrt{(f_1(N, M_y, M_z))^2 + A(f_2(T))^2}. \tag{122}$$

Based on Euler-Bernoulli beam theory and assuming circular cross-sections, we have:

- For normal stress:

$$\sigma_{xx}^N = \frac{N}{A}, \tag{123}$$

  where A is the cross-sectional area of the element.

- For bending stress:

$$\sigma_{xx}^F = -\frac{M_z y}{I_z} + \frac{M_y z}{I_y}, \tag{124}$$

  where I is the moment of inertia about the relevant axis, and y and z are the perpendicular distances to the neutral axis for M_z and M_y, respectively. Assuming a circular cross-section:

$$\sigma_{xx}^F = \pm\frac{M_r r_{ext}}{I_c}, \tag{125}$$

  where:

$$M_r = \sqrt{M_y^2 + M_z^2}, \tag{126}$$

  is the resultant moment, and I_c is the moment of inertia of the section.

- For shear stress due to torque:

$$\sigma_{xy}^T = \frac{T r_{ext}}{J_0}, \tag{127}$$

  where r_{ext} is the external radius of the cross-section, and J_0 is the polar moment of inertia.

Introducing Eqs. 123, 124, and 127 into Eq. 122, we obtain:

$$\boxed{\sigma_{eq} = \sqrt{\left(\frac{N}{A} \mp \frac{M_r r_{ext}}{I_c}\right)^2 + A\left(\frac{T r_{ext}}{J_0}\right)^2}.} \tag{128}$$

10 Augmented Lagrangian Function and Adjoint Problem

Using the same methodology as before, the Augmented Lagrangian Function along with the Adjoint Problem will be applied.

The Augmented Lagrangian Function is:

$$L_A = V(\mathbf{x}) + \frac{c}{2}\sum_{e}^{ne}\sum_{n=1}^{2}\sum_{a=0}^{1}\left\langle \frac{\mu_{e,n,a}}{c} + g_{e,n,a}\right\rangle^2 + \boldsymbol{\lambda}^T(\mathbb{K}\mathbf{U} - \mathbf{F}), \tag{129}$$

and its derivative:

$$\frac{dLA}{dx_m} = \left(\frac{\partial V}{\partial x_m} + \frac{\partial V^T}{\partial \mathbf{U}}\frac{d\mathbf{U}}{dx_m}\right) + c\sum_{e}^{ne}\sum_{n=1}^{2}\sum_{a=0}^{1}\left\langle \frac{\mu_{e,n,a}}{c} + g_{e,n,a}\right\rangle\left(\frac{\partial g_{e,n,a}}{\partial x_m} + \frac{\partial g_{e,n,a}^T}{\partial \mathbf{U}}\frac{d\mathbf{U}}{dx_m}\right) + \tag{130}$$
$$\boldsymbol{\lambda}^T\frac{d\mathbb{K}}{dx_m}\mathbf{U} + \boldsymbol{\lambda}^T\mathbb{K}\frac{d\mathbf{U}}{dx_m} - \boldsymbol{\lambda}^T\frac{d\mathbf{F}}{dx_m}.$$

Isolating terms with $\frac{d\mathbf{U}}{dx_m}$:

$$\frac{dLA}{dx_m} = \frac{\partial V}{\partial dx_m} + \boldsymbol{\lambda}^T\frac{d\mathbb{K}}{dx_m}\mathbf{U} - \boldsymbol{\lambda}^T\frac{d\mathbf{F}}{dx_m} + c\sum_{e}^{ne}\sum_{n=1}^{2}\sum_{a=0}^{1}\left\langle \frac{\mu_{e,n,a}}{c} + g_{e,n,a}\right\rangle\frac{\partial V}{\partial x_m} \tag{131}$$
$$\left[\frac{\partial V^T}{\partial \mathbf{U}} + c\sum_{e}^{ne}\sum_{n=1}^{2}\sum_{a=0}^{1}\left\langle \frac{\mu_{e,n,a}}{c} + g_{e,n,a}\right\rangle\frac{\partial V^T}{\partial \mathbf{U}} + \boldsymbol{\lambda}^T\mathbb{K}\right]\frac{d\mathbf{U}}{dx_m}.$$

Knowing that the term in brackets must equal $\mathbf{0}^T$:

$$\frac{\partial V^T}{\partial \mathbf{U}} + c\sum_{e}^{ne}\sum_{n=1}^{2}\sum_{a=0}^{1}\left\langle \frac{\mu_{e,n,a}}{c} + g_{e,n,a}\right\rangle\frac{\partial g^T}{\partial \mathbf{U}} + \boldsymbol{\lambda}^T\mathbb{K} = 0, \tag{132}$$

we find:

$$\boldsymbol{\lambda}^T\mathbb{K} = -\frac{\partial V^T}{\partial \mathbf{U}} + c\sum_{e}^{ne}\sum_{n=1}^{2}\sum_{a=0}^{1}\left\langle \frac{\mu_{e,n,a}}{c} + g_{e,n,a}\right\rangle\frac{\partial g^T}{\partial \mathbf{U}}, \tag{133}$$

which represents the adjoint problem.

Thus, we must develop the derivatives of the stress constraint. Using the equation describing the elastic deformation model:

$$\mathbb{K}(\mathbf{x})\mathbf{U}(\mathbf{x}) = \mathbf{F}(\mathbf{x}), \tag{134}$$

we know, from earlier developments, that:

$$\mathbb{K}(\mathbf{x}) = \sum_{e=1}^{ne} \mathbb{H}_e^T \mathbb{R}_e^T \underbrace{\mathbb{K}_e(\mathbf{x})}_{x_e^P \mathbb{K}_e^0} \mathbb{R}_e \mathbb{H}_e. \tag{135}$$

For each element, that is, already considering the two nodes shown in Fig. 8 in its **local** reference system,

$$\mathbf{F}_e = \mathbb{K}_e(\mathbf{x})\mathbf{u}_e, \tag{136}$$

where

$$\mathbf{F}_e = \begin{Bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \\ \vdots \\ F_{12} \end{Bmatrix}. \tag{137}$$

Thus, from Eqs. 123, 124, and 127, for **node 1**, we have that

$$\sigma_{xx} = \frac{-F_1}{A} - \frac{\sqrt{F_5^2 + F_6^2} r_e}{I_c} \tag{138}$$

and

$$\sigma_{xy} = \frac{-F_4 r_e}{J_0}. \tag{139}$$

The procedure for node 2 is analogous, but with positive signs for the corresponding nodal forces.

With this information, it is known that for sensitivity analysis, the derivative of the constraint function will be required. From Eq. 113, we have

$$\frac{dg}{dx_m} = \frac{d}{dx_m}\left(\frac{\sigma_{eq}}{\bar{\sigma}} - 1\right) = \frac{1}{\bar{\sigma}} \frac{d\sigma_{eq}}{dx_m}. \tag{140}$$

The derivative in Eq. 140 requires calculating $\frac{d\sigma_{eq}}{dx_m}$, which is given by

$$\frac{d\sigma_{eq}}{dx_m} = \frac{d}{dx_m}\left(\sqrt{\sigma_{xx}^2 + A\sigma_{xy}^2}\right) = \frac{1}{2}\frac{1}{\sigma_{eq}}\left(2\sigma_{xx}\frac{d\sigma_{xx}}{dx_m} + A2\sigma_{xy}\frac{d\sigma_{xy}}{dx_m}\right), \tag{141}$$

where

$$\frac{d\sigma_{xx}}{dx_m} = \left(-\frac{dF_1}{dx_m}\frac{1}{A} - \frac{r_e}{I_c}\frac{dM_r}{dx_m} \right), \tag{142}$$

$$\frac{\sigma_{xy}}{dx_m} = \frac{d}{dx_m}\left(\frac{-r_{ext}}{J_0} F_4 \right) = \frac{-r_{ext}}{J_0}\frac{dF_4}{dx_m}, \tag{143}$$

$$\frac{dM_r}{dx_m} = \frac{1}{2}\frac{1}{M_r}\left(2M_z \underbrace{\frac{dM_z}{dx_m}}_{\frac{-dF_6}{dx_m}} + 2M_y \underbrace{\frac{dM_y}{dx_m}}_{\frac{-dF_5}{dx_m}} \right). \tag{144}$$

It is observed that the internal forces are located in the force vector $\mathbf{F}$, which is written as

$$\mathbf{F}_e = \mathbb{K}_e \mathbf{U}_e^{local} = \mathbf{F}_{e,12\times 1}^{local} \mathbb{R}_e \mathbb{H}_e \mathbf{U}. \tag{145}$$

From now on, we will use a matrix and vectorial approach for the calculations, which will later be applied to the codes of this work.
To solve Eq. 144, we will need some definitions.
Separating by node, we have, for node 1, that the **internal forces vector** is

$$\mathbf{E}_e = [M_1]\mathbf{F}_e \tag{146}$$

and for node 2

$$\mathbf{E}_e = [M_2]\mathbf{F}_e, \tag{147}$$

where the force vector will have dimension 6×1, and the matrices M_1 and M_2 are auxiliary localization matrices, following the order N, T, My, Mz, given by

$$M_1 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{148}$$

and

$$M_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{149}$$

Thus, the internal force vectors, after extracting their respective positions (disregarding the shear forces in axes y and z), are

$$\mathbf{E}_e = \begin{Bmatrix} N \\ T \\ M_y \\ M_z \end{Bmatrix} \tag{150}$$

for nodes 1 and 2. For the modification of the variables, we will use a relaxation function, currently given by

$$fe = \rho_e^{1.5}, \tag{151}$$

directly in the element force vector,

$$\mathbf{Fe} = fe\mathbb{K}_e^0 ReHe\mathbf{U}. \tag{152}$$

Continuing,

$$\frac{d\boldsymbol{\sigma}_{e,n,a}}{dx_m} = [P_{n,a}]\frac{d\mathbf{N}}{dx_m}, \tag{153}$$

where $\frac{d\mathbf{N}}{dx_m}$, as discussed, involves all the internal forces of the element. Thus, its derivative must contain

$$\frac{dN}{dx_m} = \mathbf{L}_1^T \frac{d\mathbf{E}_n}{dx_m}, \tag{154}$$

$$\frac{dT}{dx_m} = \mathbf{L}_2^T \frac{d\mathbf{E}_n}{dx_m}, \tag{155}$$

$$\frac{dM_r}{dx_m} = \frac{1}{2}\frac{1}{M_r, n}(2\mathbf{E}_n^T[S3] + 2\mathbf{E}_n^T[S4])\frac{d\mathbf{E}_n}{dx_m}, \tag{156}$$

where $[S3]$ and $[S4]$ are auxiliary matrices that ensure the required quadratic term in the resulting moment expression in y and z.

Using another auxiliary matrix, which we will call $[D]$,

$$\frac{d\mathbf{N}}{dx_m} = \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{E_{3,n}}{M_r} & \frac{E_{4,n}}{M_r} \end{pmatrix}}_{[D]} \frac{d\mathbf{E}_n}{dx_m}. \tag{157}$$

We will also need

$$\frac{d\mathbf{E}}{dx_m} = [M_n]\frac{d\mathbf{F}}{dx_m}, \tag{158}$$

with

$$\frac{d\mathbf{F}}{dx_m} = \frac{df_e}{dx_m}\mathbb{K}_e^0\mathbb{R}_e\mathbb{H}_e\mathbf{U} + f_e\mathbb{K}_e^0\mathbb{R}_e\mathbb{H}_e\frac{d\mathbf{U}}{dx_m}. \tag{159}$$

It should be noted that, when returning the term

$$\frac{df_e}{dx_m}\mathbb{K}_e^0\mathbb{R}_e\mathbb{H}_e\mathbf{U},$$

we will have $\frac{\partial g}{\partial x_m}$, and when returning

$$f_e\mathbb{K}_e^0\mathbb{R}_e\mathbb{H}_e\frac{d\mathbf{U}}{dx_m},$$

we will have $\frac{\partial g}{\partial U}$.
Initially, we will consider that

$$f_e = x_e, \tag{160}$$

that is, the relaxation function is the element's own x. Thus,

$$\frac{df_e}{dx_m} = \begin{cases} 1, \text{ if } e = m \\ 0, \text{ if } e \neq m \end{cases}, \tag{161}$$

which, in the code, as we will always be dealing with $e = m$, means that the derivative will be 1.

10.1 Summary

The starting point of the previous section was the derivation of the Lagrange function:

$$\frac{dLA}{dx_m} = \left(\frac{\partial V}{\partial x_m} + \frac{\partial V^T}{\partial \mathbf{U}}\frac{d\mathbf{U}}{dx_m}\right) + c\sum_e^{ne}\sum_{n=1}^{2}\sum_{a=0}^{1}\left\langle\frac{\mu_{e,n,a}}{c} + g_{e,n,a}\right\rangle\left(\frac{\partial g_{e,n,a}}{\partial x_m} + \frac{\partial g_{e,n,a}^T}{\partial \mathbf{U}}\frac{d\mathbf{U}}{dx_m}\right) + \tag{162}$$
$$\boldsymbol{\lambda}^T\frac{d\mathbb{K}}{dx_m}\mathbf{U} + \boldsymbol{\lambda}^T\mathbb{K}\frac{d\mathbf{U}}{dx_m} - \boldsymbol{\lambda}^T\frac{d\mathbf{F}}{dx_m}.$$

If we initially work **only with stress constraints**, the complete optimization problem becomes

$$P\begin{cases} min\ V(\boldsymbol{\rho}) \\ S.t \\ \quad \mathbf{K}(\boldsymbol{\rho})\mathbf{U}(\boldsymbol{\rho}) = \mathbf{F} \\ \quad \sigma_{eq,e} \leq \sigma_{limiting} \\ \quad \mathbf{0} < \boldsymbol{\rho} \leq \mathbf{1} \end{cases} \tag{163}$$

11 Union with Displacement Constraints

The optimization problem can also be treated jointly with displacement constraints:

$$P\begin{cases} min\ V \\ Tq \\ |u|_i \leq \bar{u}_i, \\ \sigma_{eq,e} \leq \sigma_{esc,e} \end{cases} . \tag{164}$$

12 Uncertainties

12.1 Definitions

The problems studied so far have a deterministic approach to all variables. In other words, the values used are well-defined and consistent. However, we know that in real-world situations, this is difficult (or even impossible) to achieve. Thus, it is common for a probabilistic approach to be taken. In this way, one or more variables are studied based on certain behavior and possible deviation, and the problem is evaluated accordingly.

The first study involving uncertainties considers the **force** as an uncertain variable, i.e., it can vary with an average value and a certain variation. Let us consider that both the magnitude of the force and the angle of its application are variables. Since the force directly influences the local stress being applied, we will have an alteration in the optimization problem's constraint.

The local stress in the element will be defined as

$$\sigma = E(\sigma_j) + \beta VAR(\sigma_j), \tag{165}$$

where $E(\sigma_j)$ is the expected value of the stress, β is a positive constant (the higher it is, the lower the probability of failure), and $VAR(\sigma_j)$ is the variance of the local stress.

Thus, the updated optimization problem is

$$P\begin{cases} min\ V(\boldsymbol{\rho}) \\ S.t \\ \quad \mathbf{K}(\boldsymbol{\rho})\mathbf{U}(\boldsymbol{\rho}) = \mathbf{F} \\ \quad E(\sigma_j) + \beta VAR(\sigma_j) \leq \bar{\sigma} \\ \quad \mathbf{0} < \boldsymbol{\rho} \leq \mathbf{1} \end{cases} . \tag{166}$$

It can be observed that if $\beta \to 0$, the problem becomes identical to the one we had previously.

The constraints, previously defined in the deterministic approach as

$$g_j^D = \frac{\sigma_j}{\bar{\sigma}} - 1, \tag{167}$$

become, in the probabilistic approach,

$$g_j^P = \frac{E(\sigma_j) + \beta VAR(\sigma_j)}{\bar{\sigma}} - 1 = \frac{E(\sigma_j)}{\bar{\sigma}} + \frac{\beta}{\bar{\sigma}} VAR(\sigma_j) - 1. \tag{168}$$

It is known that for sensitivity analysis, it is necessary to calculate the derivative of the constraint function. Therefore, we have that

$$\frac{dg_j^P}{dx_m} = \frac{1}{\bar{\sigma}} \frac{dE(\sigma_j)}{dx_m} + \frac{\beta}{\bar{\sigma}} \frac{dVAR(\sigma_j)}{dx_m}. \tag{169}$$

In other words, we need both the derivative of the expected value and the variance of the local stress. These, along with the expected value and variance of the function itself, will be calculated using an algorithm developed as an adaptation of the Monte Carlo Algorithm, the Local Averaged Stratified Sampling (LASS).

12.2 A Global Approach to Stress Constraints

However, even using LASS, where calculations are performed through a central value in the generated bins, we will still have many constraints in the problem, and the program may not be sufficiently optimized. Thus, an alternative is to use the maximum value of the equivalent stress in each element. That is, the optimization problem is reformulated as

$$P\begin{cases} min\ V(\boldsymbol{\rho}) \\ S.t \\ \quad \mathbf{K}(\boldsymbol{\rho})\mathbf{U}(\boldsymbol{\rho}) = \mathbf{F} \\ \quad \max(\sigma_j) \leq \bar{\sigma} \\ \quad \mathbf{0} < \boldsymbol{\rho} \leq \mathbf{1} \end{cases} . \tag{170}$$

That is, we can use a **global** strategy to minimize the computational cost required.

Initially, we can use the approach of Chao-le, which uses the P norm of the stress constraint. The P norm is given by

$$||\boldsymbol{\sigma}||_P = \left(\sum_{i=1}^{n} \boldsymbol{\sigma}_i^P \right)^{\frac{1}{P}}, \tag{171}$$

where p is a positive constant, preferably even. As $p \to \infty$, the largest value inside the summation will dominate the others, and $||\boldsymbol{\sigma}||_p \to \max(\sigma)$. The way to interpret this consideration was proposed by Chao-le, where the constraint is understood as

$$\frac{\max(\boldsymbol{\sigma}^{k-1})}{||\boldsymbol{\sigma}||_p^{k-1}} ||\boldsymbol{\sigma}||_p^k \leq \bar{\sigma}. \tag{172}$$

Thus, according to the understanding that, as the optimization process progresses, changes in the structure and the variables involved become smaller, $||\boldsymbol{\sigma}||_p^{k-1} \to ||\boldsymbol{\sigma}||_p^k$ and, therefore,

$$\frac{||\boldsymbol{\sigma}||_p^k}{||\boldsymbol{\sigma}||_p^{k-1}} \to 1 \tag{173}$$

and we get

$$\max(\boldsymbol{\sigma}^{k-1}) 1 \leq \bar{\sigma}. \tag{174}$$

Let us define

$$\frac{\max(\boldsymbol{\sigma}^{k-1})}{||\boldsymbol{\sigma}||_p^{k-1}} = c^{k-1}, \tag{175}$$

so that the stress constraint becomes

$$g = \frac{c^{k-1} ||\boldsymbol{\sigma}||_p^k}{\bar{\sigma}} - 1. \tag{176}$$

The Augmented Lagrange Function with the Adjoint Problem is

$$LA^k(\rho_m) = \mathbf{V}(\rho_m) + \frac{r}{2} \left\langle \frac{\mu}{r} + \frac{c^{k-1} ||\boldsymbol{\sigma}_{eq}||_p}{\bar{\sigma}} - 1 \right\rangle^2 + \boldsymbol{\lambda}^T (\mathbb{K}\mathbf{U} - \mathbf{F}) \tag{177}$$

and its gradient is

$$\frac{dLA^k(\rho_m)}{d\rho_m} = \frac{d\mathbf{V}(\rho_m)}{d\rho_m} + r\left\langle \frac{\mu}{r} + \frac{c^{k-1}||\sigma_{eq}||_P}{\bar{\sigma}} - 1 \right\rangle \frac{c^{k-1}}{\bar{\sigma}} \frac{1}{P} \left(\sum_{i=1}^{ne} \sigma_{eq,i}^P \right)^{\frac{1}{P}-1} \sum_{i=1}^{4ne} P\sigma_{eq,i}^{P-1} \frac{d\sigma_{eq,i}}{d\rho_m} + \boldsymbol{\lambda}^T \frac{d\mathbb{K}}{d\rho_m}\mathbf{U} + \boldsymbol{\lambda}^T \mathbb{K} \frac{d\mathbf{U}}{d\rho_m} - \boldsymbol{\lambda}^T \frac{d\mathbf{F}}{d\rho_m}. \tag{178}$$

Where

$$\frac{d\sigma_{eq,i}}{d\rho_m} = \frac{\partial \boldsymbol{\sigma}_i}{\partial \rho_m} + \frac{\partial \boldsymbol{\sigma}_i^T}{\partial \mathbf{U}} \frac{d\mathbf{U}}{d\rho_m}, \tag{179}$$

the derivative of the Augmented Lagrange Function becomes

$$\frac{dLA^k(\rho_m)}{d\rho_m} = \frac{d\mathbf{V}(\rho_m)}{d\rho_m} + r\left\langle \frac{\mu}{r} + \frac{c^{k-1}||\sigma_{eq}||_P}{\bar{\sigma}} - 1 \right\rangle \frac{c^{k-1}}{\bar{\sigma}} \frac{1}{P} \left(\sum_{e=1}^{ne} \sum_{no=1}^{2} \sum_{i=0}^{1} \sigma_{eq,i}^P \right)^{\frac{1}{P}-1} \sum_{e=1}^{ne} \sum_{no=1}^{2} \sum_{i=0}^{1} P\sigma_{eq,i}^{P-1} \left(\frac{\partial \boldsymbol{\sigma}_i}{\partial \rho_m} + \frac{\partial \boldsymbol{\sigma}_i^T}{\partial \mathbf{U}} \frac{d\mathbf{U}}{d\rho_m} \right) + \boldsymbol{\lambda}^T \frac{d\mathbb{K}}{d\rho_m}\mathbf{U} + \boldsymbol{\lambda}^T \mathbb{K} \frac{d\mathbf{U}}{d\rho_m} - \boldsymbol{\lambda}^T \frac{d\mathbf{F}}{d\rho_m}. \tag{180}$$

Knowing that $\mathbf{F}$ does not depend on the design variables,

$$\frac{d\mathbf{F}}{d\rho_m} = \mathbf{0}. \tag{181}$$

With the terms involving $\frac{d\mathbf{U}}{d\rho_m}$ highlighted, we get

$$\frac{dLA^k(\rho_m)}{d\rho_m} = \frac{d\mathbf{V}(\rho_m)}{d\rho_m} + \left[r\left\langle \frac{\mu}{r} + \frac{c^{k-1}||\sigma_{eq}||_P}{\bar{\sigma}} - 1 \right\rangle \frac{c^{k-1}}{\bar{\sigma}} \frac{1}{P} \left(\sum_{e=1}^{ne} \sum_{no=1}^{2} \sum_{i=0}^{1} \sigma_{eq,i}^P \right)^{\frac{1}{P}-1} \sum_{e=1}^{ne} \sum_{no=1}^{2} \sum_{i=0}^{1} \left(P\sigma_{eq,i}^{P-1} \left(\frac{\partial \sigma_{eq,i}^T}{\partial \mathbf{U}} \right) \right) + \boldsymbol{\lambda}^T \mathbb{K} \right] \frac{d\mathbf{U}}{d\rho_m} + r\left\langle \frac{\mu}{r} + \frac{c^{k-1}||\sigma_{eq}||_P}{\bar{\sigma}} - 1 \right\rangle \frac{c^{k-1}}{\bar{\sigma}} \frac{1}{P} \left(\sum_{e=1}^{ne} \sum_{no=1}^{2} \sum_{i=0}^{1} \sigma_{eq,i}^P \right)^{\frac{1}{P}-1} \sum_{e=1}^{ne} \sum_{no=1}^{2} \sum_{i=0}^{1} \left(P\sigma_{eq,i}^{P-1} \right) \frac{\partial \boldsymbol{\sigma}_i}{\partial \rho_m} + \frac{\partial \boldsymbol{\sigma}_i^T}{\partial \mathbf{U}} \frac{d\mathbf{U}}{d\rho_m} + \boldsymbol{\lambda}^T \frac{d\mathbb{K}}{d\rho_m}\mathbf{U}. \tag{182}$$

Knowing that the term in brackets must result in zero, we have the adjoint problem,

$$\boldsymbol{\lambda}^T\mathbb{K} = -r\left\langle \frac{\mu}{r} + \frac{c^{k-1}||\sigma_{eq}||_P}{\bar{\sigma}} - 1 \right\rangle \frac{c^{k-1}}{\bar{\sigma}} \frac{1}{P} \left(\sum_{e=1}^{ne} \sum_{no=1}^{2} \sum_{i=0}^{1} \sigma_{eq,i}^P \right)^{\frac{1}{P}-1} \sum_{e=1}^{ne} \sum_{no=1}^{2} \sum_{i=0}^{1} \left[P\sigma_{eq,i}^{P-1} \left(\frac{\partial \sigma_{eq,i}^T}{\partial \mathbf{U}} \right) \right], \tag{183}$$

where

$$\frac{\partial \sigma_{eq,i}^T}{\partial \mathbf{U}} = fe\mathbb{K}_e\mathbb{R}_e\mathbb{H}_e \frac{d\mathbf{U}}{d\rho_m}. \tag{184}$$

It is important to note that fe is the relaxation function, and according to Matteo Bruggi's qp approach,

Following the same pattern as the development of previous codes, the adjoint loading vector will be

$$D_s = r\left\langle \frac{\mu}{r} + \frac{c||\sigma_{eq}||_P}{\bar{\sigma}} - 1 \right\rangle \frac{c}{\bar{\sigma}} \frac{1}{P} \left(\sum_{e=1}^{ne} \sum_{no=1}^{2} \sum_{i=0}^{1} \sigma_{eq,i}^P \right)^{\frac{1}{P}-1} \sum_{e=1}^{ne} \sum_{no=1}^{2} \sum_{i=0}^{1} \left(P\sigma_{eq,i}^{P-1} \left(fe\mathbb{K}_e\mathbb{R}_e\mathbb{H}_e \frac{d\mathbf{U}}{d\rho_m} \right) \right). \tag{185}$$

In the code,

$$r\frac{c}{\bar{\sigma}}\frac{1}{P} = \text{constante}, \tag{186}$$

$$\left\langle \frac{\mu}{r} + \frac{c||\sigma_{eq}||_P}{\bar{\sigma}} - 1 \right\rangle = \text{heav} \tag{187}$$

$$\left(\sum_{e=1}^{ne} \sum_{no=1}^{2} \sum_{i=0}^{1} \sigma_{eq,i}^P \right)^{\frac{1}{P}-1} = \text{soma1} \tag{188}$$

$$\sum_{e=1}^{ne} \sum_{no=1}^{2} \sum_{i=0}^{1} \left(P\sigma_{eq,i}^{P-1} \right) \quad \left(fe\mathbb{K}_e\mathbb{R}_e\mathbb{H}_e \frac{d\mathbf{U}}{d\rho_m} \right). \tag{189}$$