

nto



## **aula 10 / Otimização estrutural em engenharia**

190924



## Sumário (TP+P)

Otimização estrutural em Engenharia  
Definição e classificação  
Overview

Resolução dos problemas black-box  
Exemplos

Otimização com recurso a programas FEA  
Caso do Abaqus  
Caso do SolidWorks



<https://forms.office.com/e/fR29HVDxs2>



# O que é Otimização Estrutural?

Otimização estrutural é o processo de encontrar a melhor configuração para uma estrutura que atenda a critérios específicos.

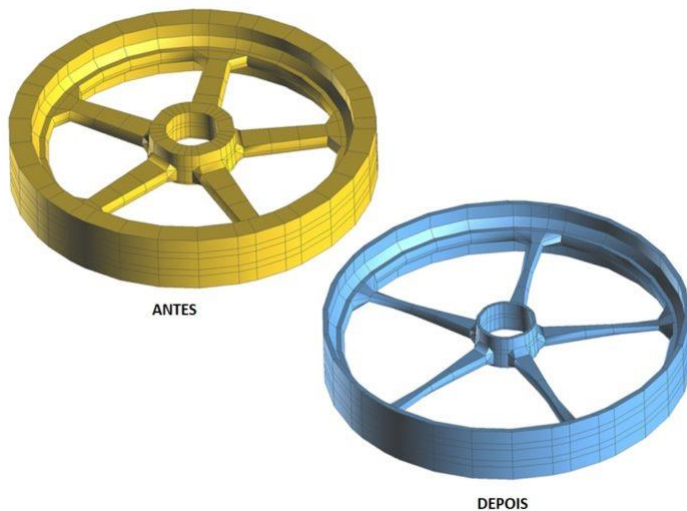
- Objetivo: Minimizar peso, custo, ou material enquanto se mantém resistência, rigidez e segurança.
- Aplicações: Usada em diversas áreas como construção civil, engenharia aeronáutica, e mecânica.



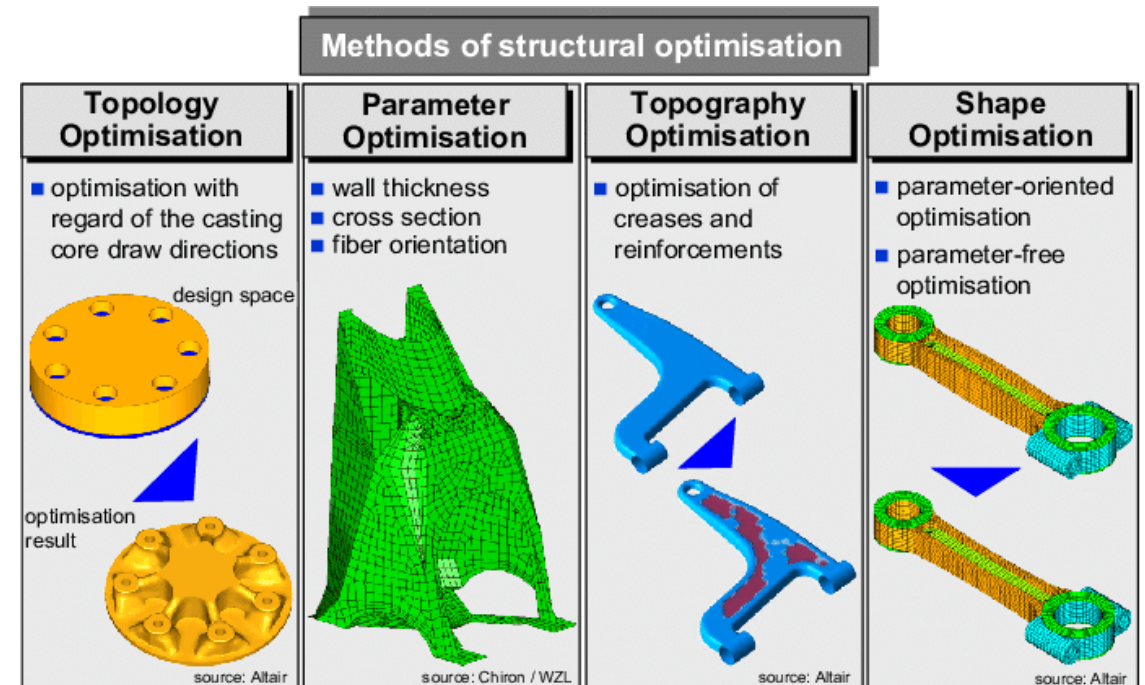


# Tipos de Otimização Estrutural

1. Otimização de Forma: Ajuste da forma da estrutura para otimizar desempenho.
2. Otimização Topológica: Determinação da melhor distribuição de material no espaço.
3. Otimização Paramétrica: Ajuste de parâmetros como espessura ou diâmetro.
4. Otimização Multidisciplinar: Integração de otimização estrutural com outros critérios como aerodinâmica ou custo.



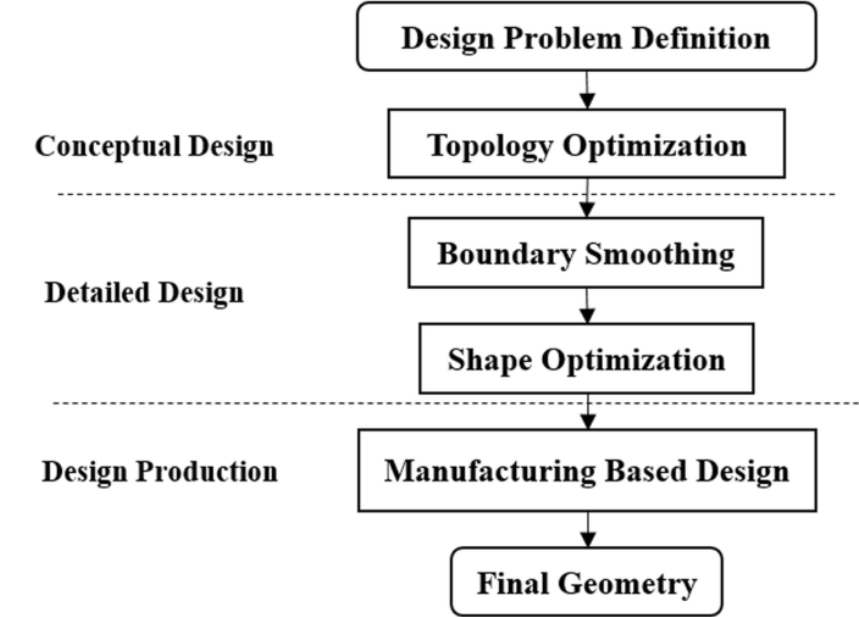
Otimização paramétrica.





# Processo de Otimização Estrutural

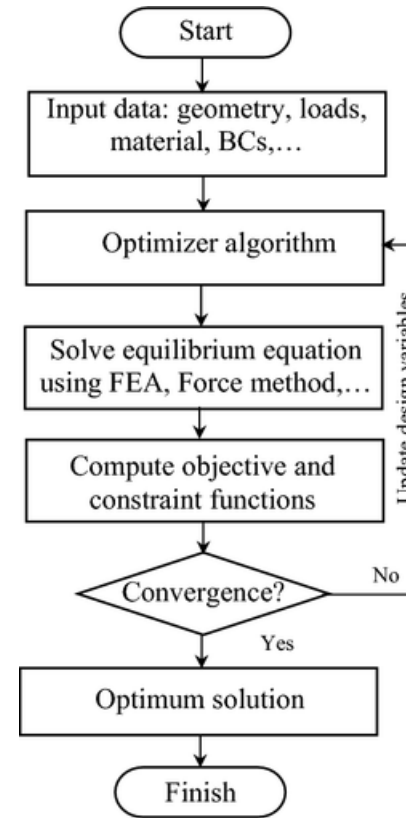
1. Definir o Problema: Identificar o objetivo (ex.: minimizar peso) e as restrições (ex.: resistência mínima).
2. Escolher Variáveis de Projeto: Parâmetros que podem ser alterados (ex.: espessura, comprimento).
3. Definir Função Objetivo e Restrições: Criar uma função que quantifica o desempenho (ex.: peso total).
4. Selecionar o Método de Otimização: Escolher o algoritmo adequado (ex.: SLSQP, otimização topológica).
5. Executar e Analisar os Resultados: Avaliar a solução obtida e ajustá-la conforme necessário.



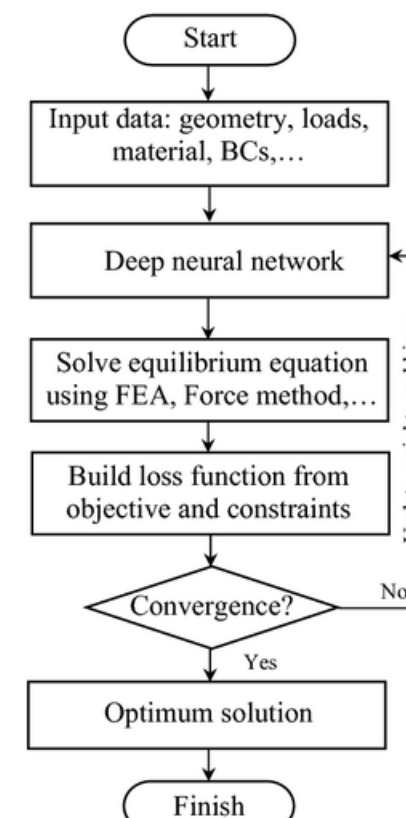


# Métodos de Otimização

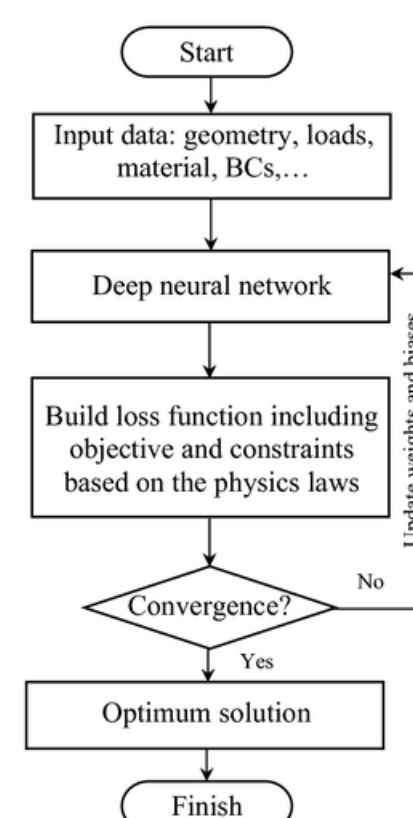
- Métodos Determinísticos:
  - Gradiente: Busca o mínimo usando derivadas.
  - SLSQP (Sequential Least Squares Quadratic Programming): Bom para problemas com restrições.
- Métodos Evolutivos e Algoritmos Genéticos:
  - Inspirados na evolução natural, úteis para problemas complexos e não lineares.
- Métodos Heurísticos:
  - Ex.: Algoritmo das Partículas (PSO), Simulated Annealing. Bons para encontrar soluções em espaços de busca grandes.



(a)



(b)



(c)



## Exemplo Prático - Otimização de uma Viga

- Problema: Minimizar o peso de uma viga sujeita a uma carga específica.
- Variáveis de Projeto: Altura e largura da viga.
- Restrições: Tensão máxima e deflexão.
- Resultado: Otimização da geometria para garantir menor peso e suficiente resistência.



**KEEP  
CALM  
AND**

**ENJOY YOUR  
LAST CLASS**



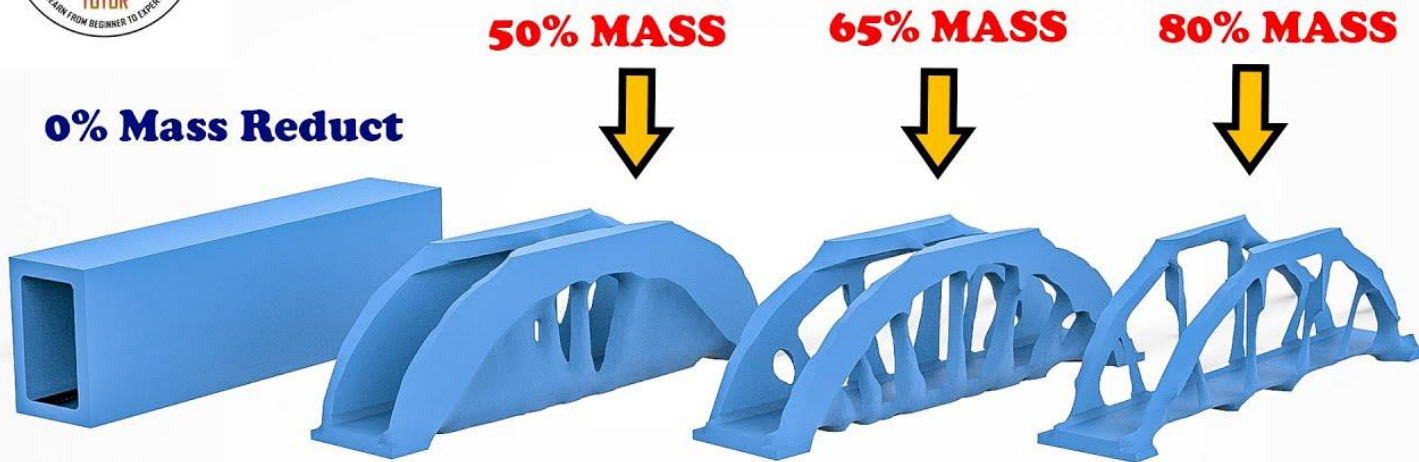


# Ferramentas de Software para Otimização Estrutural

- SolidWorks: Com módulos de FEA para simulações estruturais.
- ANSYS: Avançado para análise estrutural e otimização topológica.
- MATLAB: Extensivamente usado para otimizações baseadas em algoritmos personalizados.
- Python (scipy.optimize): Ideal para customizar otimizações com bibliotecas como scipy e numpy.



## SOLIDWORKS SIMULATION TOPOLOGY OPTIMIZATION

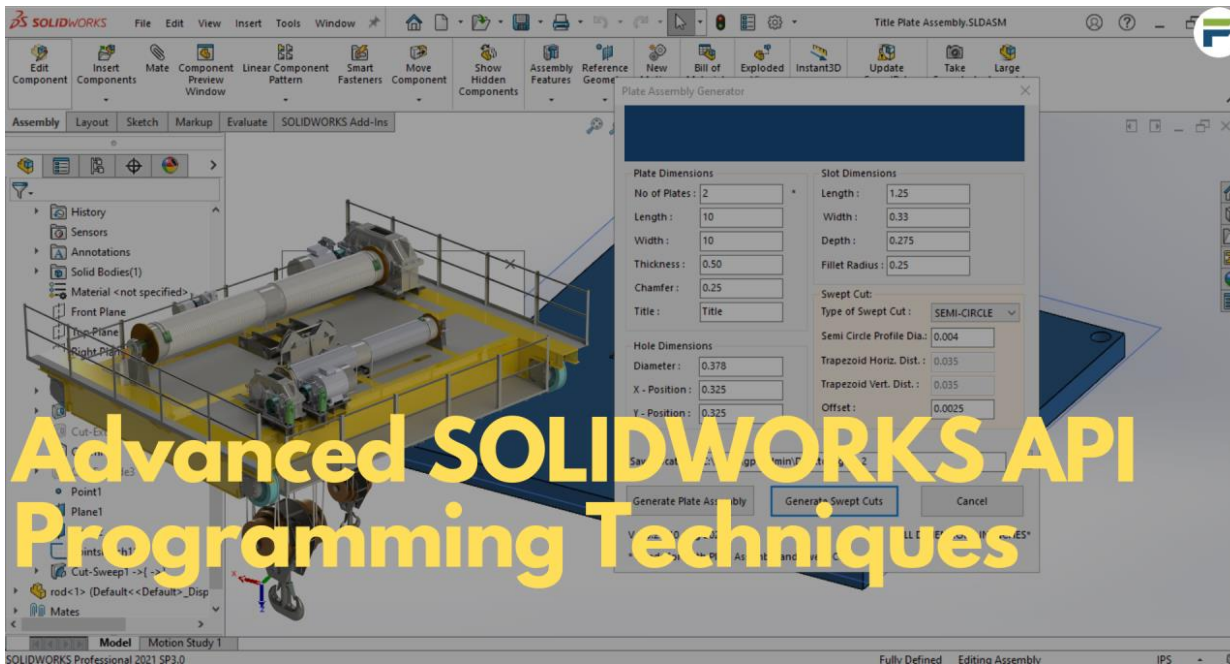


**BEST STIFNESS TO WEIGHT RATIO**



# Integração de Ferramentas - Python com SolidWorks

- Objetivo: Controlar o SolidWorks a partir do Python para realizar simulações de otimização.
- Método: Uso da API do SolidWorks para modificar parâmetros e extrair resultados.
- Exemplo: Minimização de peso ajustando parâmetros de diâmetro e espessura de uma peça.



<https://www.enggtechnique.com/resource/blog-detail/solidworks-api-how-it-works>

<https://www.linkedin.com/pulse/exploring-beyond-basics-advanced-solidworks-api-programming-kora/>



# Desafios na Otimização Estrutural

- Convergência:  
Dificuldade em encontrar o ponto ótimo em problemas complexos.
- Tempo de Computação:  
Simulações podem ser demoradas, especialmente para FEA detalhado.
- Restrição de Projeto:  
Limitações de fabricação e custos devem ser consideradas.
- Integração Multidisciplinar:  
Coordenação com outras áreas, como dinâmica de fluidos ou térmica.







# Futuro da Otimização Estrutural

- Benefícios: Otimização estrutural permite projetos mais eficientes, leves e sustentáveis.
- Tendências Futuras:
  - Uso crescente de inteligência artificial e machine learning em otimização.
  - Desenvolvimento de materiais novos que permitam novas abordagens.
  - Integração de métodos de fabricação avançada, como impressão 3D.





Resolução de problemas de otimização

# Resolução de problemas

...cuj a função-objetivo é calculada por programas externos, como é o caso da otimização estrutural!



## Otimização black-box

Muitas vezes a função-objetivo é avaliada por programas cujo interior é inacessível



Função-objetivo do  
tipo *black-box*





# Otimização black-box

Nesse caso,





# Otimização black-box

- Otimização *black-box*-----





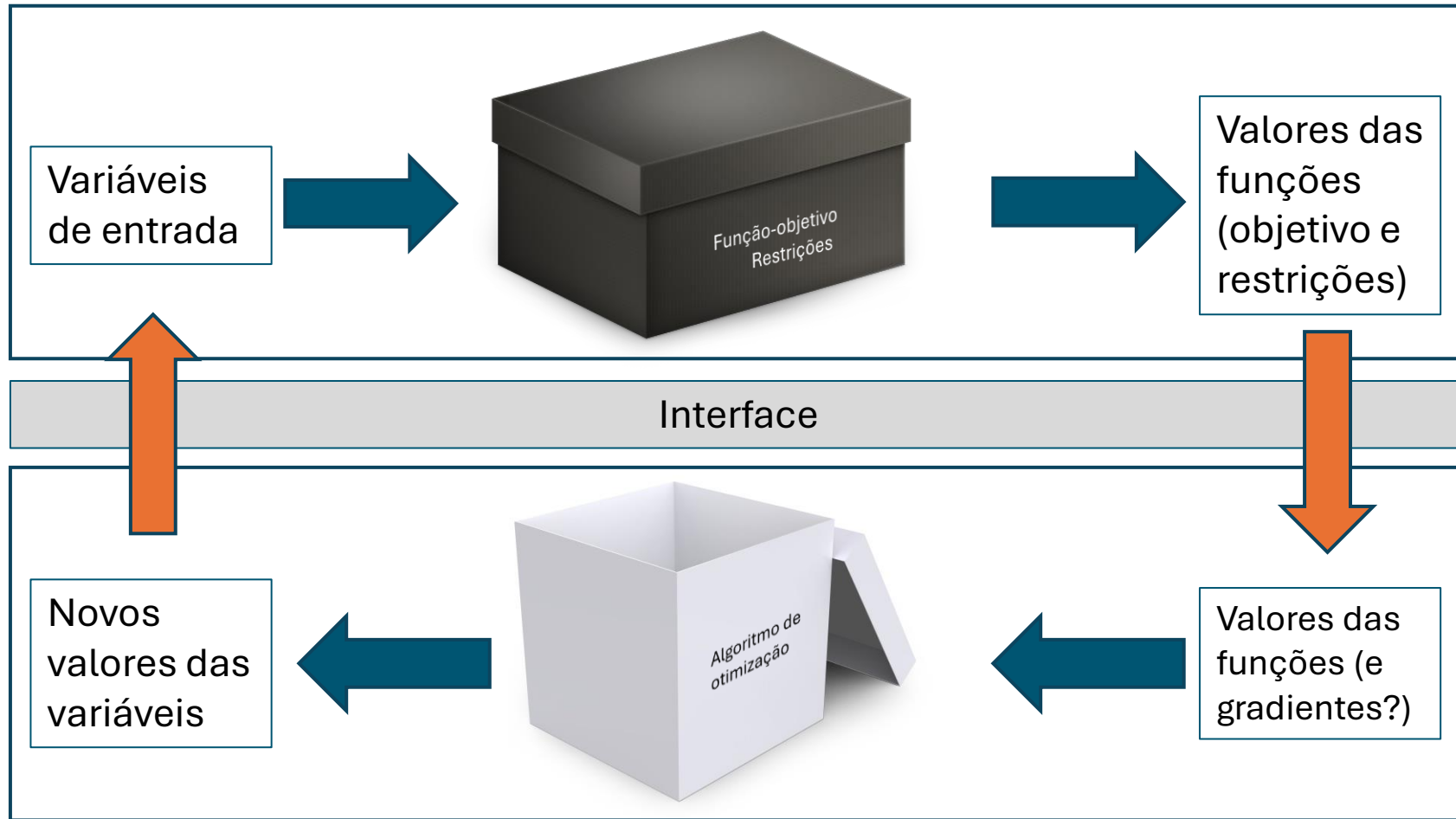


# Otimização black-box





# Otimização black-box





# Resolução de problemas

...cuja função-objetivo é calculada por programas externos!

## Parte prática...



# Otimização black-box

**Exemplo:**

**Programa de otimização:** Excel (*Solver*)

**Funções:** programa externo.

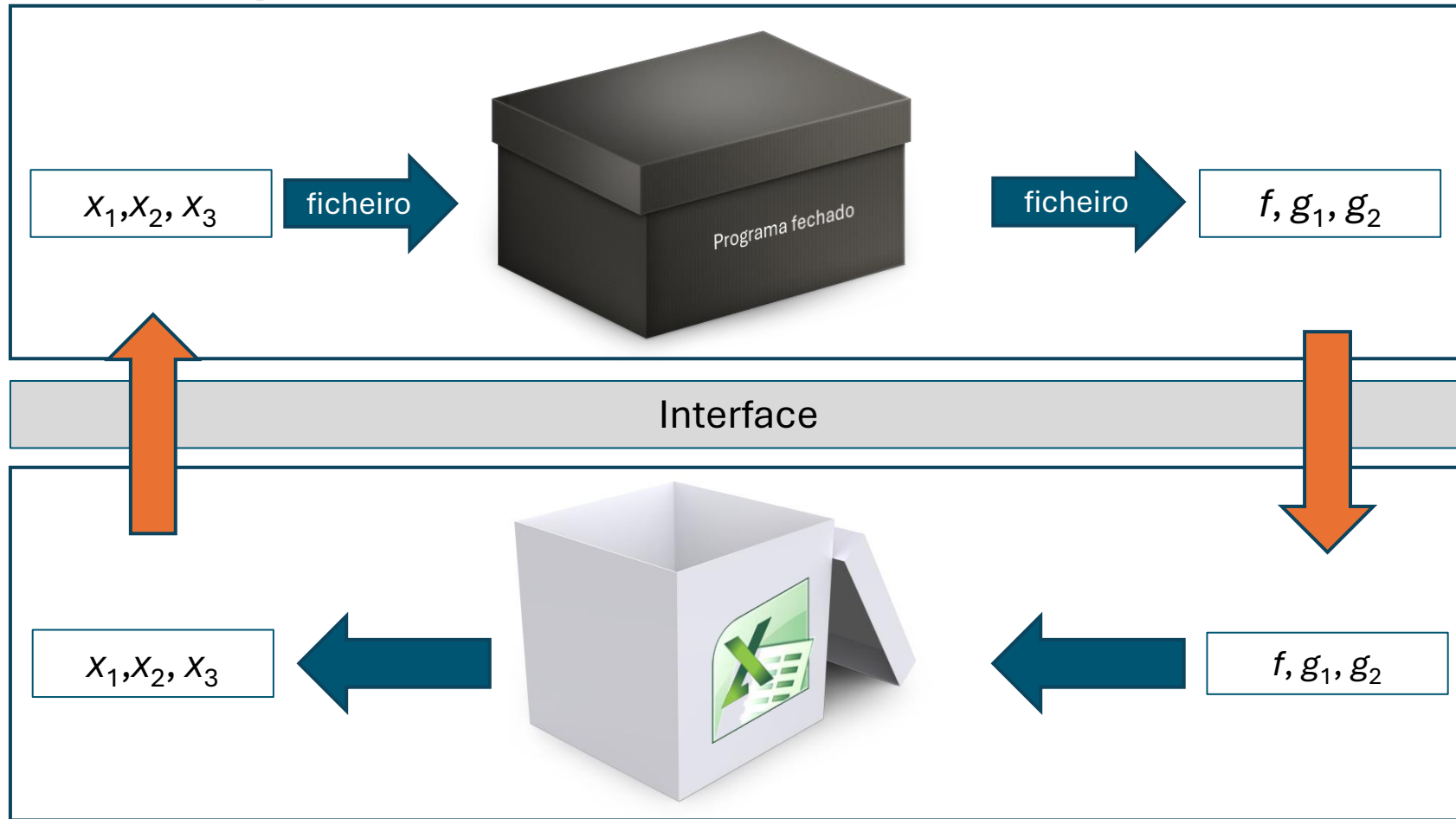
**Interface:** VBA (*Visual Basic for Applications*)

**Problema:**

$$\begin{aligned} \text{maximizar } & f(\mathbf{x}) = x_1x_2 + x_2x_3, \\ \text{sujeito a } & g_1(\mathbf{x}) = x_1^2 - x_2^2 + x_3^2 \leq 2, \\ & g_2(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2 \leq 10. \end{aligned}$$



# Otimização black-box





# Otimização black-box

## Programa externo (exemplo em Fortran):

```
program ProgramaExterior
  Real*8 x1, x2, x3, fobj, g1, g2
  character*62 FilePath
  character*72 file1
  character*79 file2
  character*78 file3

  ! Abertura e leitura do ficheiro com as variáveis de optimização
  FilePath='C:\Users\'
  file1=FilePath // "OptVar.txt"
  Open(unit=23,file=file1)
    read(23,*)x1
    read(23,*)x2
    read(23,*)x3
  Close (23)

  ! Cálculo da função objectivo e funções restrição
  fobj = x1*x2 + x2*x3
  g1   = x1**2. - x2**2. + x3**2.
  g2   = x1**2. + x2**2. + x3**2.

  ! Abertura e escrita do ficheiro com os valores da função objectivo e restrições
  file2=FilePath // "ObjFuncResult.txt"
  Open(unit=24,file=file2)
    write(24,*)fobj
  Close (24)

  file3=FilePath // "Restr1Result.txt"
  Open(unit=25,file=file3)
    write(25,*)g1
  Close (25)

  file3=FilePath // "Restr2Result.txt"
  Open(unit=26,file=file3)
    write(26,*)g2
  Close (26)

end program
```

$$\begin{aligned} &\text{maximizar } f(\mathbf{x}) = x_1x_2 + x_2x_3, \\ &\text{sujeito a } g_1(\mathbf{x}) = x_1^2 - x_2^2 + x_3^2 \leq 2, \\ &\quad \quad g_2(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2 \leq 10. \end{aligned}$$

Ficheiros



# Otimização black-box

## Programa externo (exemplo em Python):

```
# This is an external program
from pathlib import Path
#print('Hello, world! External program running!')

data_folder = Path("C:/Users/gilac/Desktop/ExemploVBA-Excel/")
# Abertura e leitura do ficheiro com as variáveis de optimização
filepath= data_folder / 'OptVar.txt'

with open(filepath,'r') as fp:
    line = fp.readline()
    x1 = float(line)
    line = fp.readline()
    x2 = float(line)
    line = fp.readline()
    x3 = float(line)
    cnt = 1
    fp.close()
#print('x1=',x1,' x2=',x2,' x3=',x3)

# Cálculo da função objectivo e funções restrição
fobj = x1*x2 + x2*x3
g1 = x1**2. - x2**2. + x3**2.
g2 = x1**2. + x2**2. + x3**2.

#print('fobj=',fobj, ' g1=', g1,'?<=2; g2=', g2,'?<=10')

# Abertura e escrita do ficheiro com os valores da função objectivo e restrições
filepath=data_folder / 'ObjFuncResult.txt'
file1=open(filepath, 'w')
file1.write(str(fobj))
file1.close()

filepath=data_folder / 'Restr1Result.txt'
file2=open(filepath, 'w')
file2.write(str(g1))
file2.close()

filepath=data_folder / 'Restr2Result.txt'
file3=open(filepath, 'w')
file3.write(str(g2))
file3.close()
```

$$\begin{aligned} &\text{maximizar } f(\mathbf{x}) = x_1x_2 + x_2x_3, \\ &\text{sujeito a } g_1(\mathbf{x}) = x_1^2 - x_2^2 + x_3^2 \leq 2, \\ &\quad \quad g_2(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2 \leq 10. \end{aligned}$$

Ficheiros



# Otimização black-box

## Programa externo (exemplo em Matlab):

```
% This is an external program
% Abertura e leitura do ficheiro com as variáveis de optimização
file=fopen('OptVar.txt','r');
vars=fscanf(file,'%f');
fclose(file)

x1=vars(1,1);
x2=vars(2,1);
x3=vars(3,1);

% Cálculo da função objectivo e funções restrição
fobj=x1*x2+x2*x3;
g1=(x1^2)-(x2^2)+(x3^2);
g2=(x1^2)+(x2^2)+(x3^2);

% Abertura e escrita do ficheiro com os valores da função objectivo e restrições
new_file1=fopen('ObjFuncResult.txt','w');
fprintf(new_file1,'%u',fobj)
fclose(new_file1)

new_file2=fopen('Restr1Result.txt','w');
fprintf(new_file2,'%u',g1)
fclose(new_file2)

new_file3=fopen('Restr2Result.txt','w');
fprintf(new_file3,'%u',g2)
fclose(new_file3)
```

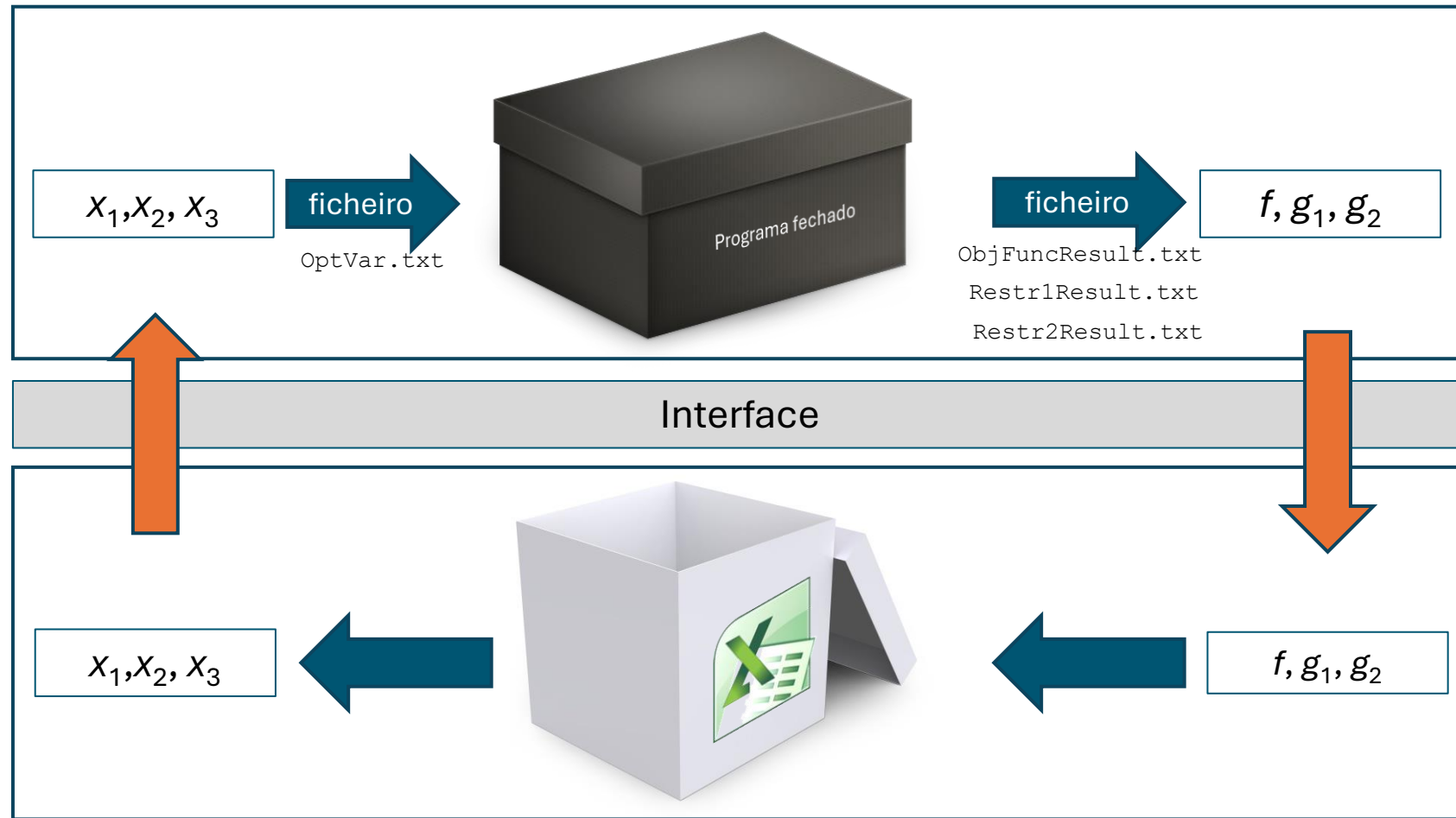
maximizar  $f(\mathbf{x}) = x_1x_2 + x_2x_3$ ,  
sujeito a  $g_1(\mathbf{x}) = x_1^2 - x_2^2 + x_3^2 \leq 2$ ,  
 $g_2(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2 \leq 10$ .

Ficheiros





# Otimização black-box





# Otimização com excel

## Interface: VBA (no Excel)

### 1 Activar as macros

Ativar as macros e guardar o ficheiro Excel num ficheiro preparado para a execução de macros (formato xlsx). Para ativar a funcionalidade de macros, basta ir às opções do programa Excel (Ficheiro - Opções - Centro de Fidedignidade - Definições de Macros).

### 2 Aceder ao modo de programador

Por defeito, este modo de programador não se encontra acessível. Deste modo, sugere-se ir a Ficheiro - Opções - Personalizar Friso - Na direita seleccionar Programador. Este procedimento adiciona um novo menu no programa Excel chamado Programador. Neste, estão acessíveis os acessos ao Visual Basic, Macros, etc.

### 3 Criar uma função VBA

Após o acesso ao Visual Basic (Menu - Programador - Visual Basic), deve-se clicar com o botão direito do rato na folha ativa do Excel (por defeito, esta chama-se "Folha 1"). No menu que aparece, seleccionar Insert - Module. A janela criada serve para a inserção do código em VBA com as funções e sub-rotinas.

Office 2010





# Otimização black-box

## Interface: VBA (no Excel)

```
'Obriga à declaração de todas as variáveis
Option Explicit
Public Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)

Function OptFunc(x1, x2, x3)
    Dim PathName As String
    Dim VarFileName As String
    Dim ObjFuncFileName As String
    Dim file2Write As Integer
    Dim file2Read As Integer
    Dim ObjFuncResult
    Dim RetVal

    'Abertura do ficheiro para escrita das variáveis de optimização
    PathName = "C:\Users\"
    VarFileName = PathName & "OptVar.txt"
    file2Write = FreeFile()
    Open VarFileName For Output As file2Write
        Print #file2Write, x1
        Print #file2Write, x2
        Print #file2Write, x3
    Close #file2Write

    'Execução do programa exterior
    ' Chamando um código executável (em Fortran)
    'RetVal = Shell(PathName & "ProgramaExterior.exe", vbNormalFocus)
    ' Chamando um código em python
    Shell "cmd /c" & PathName & "ProgramaExterior.py", vbHide
    Sleep 600 'tempo necessário para a execução completa do programa

    'Abertura do ficheiro para leitura do resultado da função objectivo
    ObjFuncFileName = PathName & "ObjFuncResult.txt"
    file2Read = FreeFile()
    Open ObjFuncFileName For Input As #file2Read
        Input #file2Read, ObjFuncResult
    Close #file2Read
    OptFunc = ObjFuncResult

End Function
```

"C:\Program  
Files\MATLAB\R2020a\bin\matlab.ex  
e" -nodisplay -nosplash -nodesktop -  
r "run('C:\Users\xxx\Desktop\Nova  
pasta (3)\vba.m'); exit;"

Ficheiros



# Otimização black-box

## Interface: VBA (no Excel)

```
Function Restr1(x1, x2, x3)
    Dim PathName As String
    Dim RestrFileName As String
    Dim file2Read As Integer
    Dim RestrResult1

    'Abertura do ficheiro para leitura do resultado da função restrição
    PathName = "C:\Users\"
    RestrFileName = PathName & "Restr1Result.txt"
    file2Read = FreeFile()
    Open RestrFileName For Input As #file2Read
        Input #file2Read, RestrResult1
    Close #file2Read
    Restr1 = RestrResult1
End Function
```

```
Function Restr2(x1, x2, x3)
    Dim PathName As String
    Dim RestrFileName As String
    Dim file2Read As Integer
    Dim RestrResult2

    'Abertura do ficheiro para leitura do resultado da função restrição
    PathName = "C:\Users\"
    RestrFileName = PathName & "Restr2Result.txt"
    file2Read = FreeFile()
    Open RestrFileName For Input As #file2Read
        Input #file2Read, RestrResult2
    Close #file2Read
    Restr2 = RestrResult2
End Function
```

Ficheiros





# Otimização black-box

## Programa de otimização: Excel

	A	B	C	D
1	Problema cuja função é calculada por programa externo			
2				
3	x=	1		
4		2		
5		1		
6				
7	f=	=OptFunc(B3:B4;B5)		
8				
9	g 1=	=Restr1(B3;B4;B5)+B7-B7	<=	2
10	g 2=	=Restr2(B3;B4;B5)+B7-B7	<=	10

- A inserção da adição e da subtração do valor da célula B7 nada contribui para o resultado.
- Isto obriga a que a célula B7, que contém a função *Optfunc*, seja calculada primeiro.
- Assim, a leitura dos ficheiros *Restr1Result.txt* e *Restr2Result.txt* é feita sempre depois da execução do programa externo.



# Otimização black-box

## Programa de otimização: Excel

	A	B	C	D
1	Problema cuja função é calculada por programa externo			
2				
3	x=	1		
4		2		
5		1		
6				
7	f=	=OptFunc(B3:B4;B5)		
8				
9	g 1=	=Restr1(B3;B4;B5)+B7-B7	<=	2
10	g 2=	=Restr2(B3;B4;B5)+B7-B7	<=	10

- A inserção da adição e da subtração do valor da célula B7 nada contribui para o resultado.
- Isto obriga a que a célula B7, que contém a função *Optfunc*, seja calculada primeiro.
- Assim, a leitura dos ficheiros *Restr1Result.txt* e *Restr2Result.txt* é feita sempre depois da execução do programa externo.



# Otimização black-box

## Programa de otimização: Excel



## ***Black-box problems***

Extras?





## Black-box problems: using FEA codes (e.g. Abaqus)

More complicated problems can be simulated numerically by using commercial general **finite element** softwares, such as **Abaqus**

Abaqus allows the user to create **parametric input files**



## Black-box problems: using FEA codes (e.g. Abaqus)

More complicated problems can be simulated numerically by using commercial general **finite element** softwares, such as **Abaqus**

Abaqus allows the user to create **parametric input files**

```
*NODE
0.0, 0.0,
1.0, 0.0,
1.0, 1.0,
0.0, 1.0
*ELEMENT, TYPE=CPS4
1,2,4,3
```



## Black-box problems: using FEA codes (e.g. Abaqus)

More complicated problems can be simulated numerically by using commercial general **finite element** softwares, such as **Abaqus**

Abaqus allows the user to create **parametric input files**

```
*NODE
```

```
0.0, 0.0,  
1.0, 0.0,  
1.0, 1.0,  
0.0, 1.0
```

**Node data**

```
*ELEMENT, TYPE=CPS4
```

```
1, 2, 4, 3
```

**Element data**

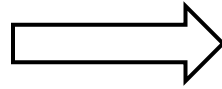


## Black-box problems: using FEA codes (e.g. Abaqus)

More complicated problems can be simulated numerically by using commercial general **finite element** softwares, such as **Abaqus**

Abaqus allows the user to create **parametric input files**

```
*NODE
0.0, 0.0,
1.0, 0.0,
1.0, 1.0,
0.0, 1.0
*ELEMENT, TYPE=CPS4
1,2,4,3
```



```
*Parameter
x1=1.0
y1=1.0
***
```

**Parameter definition**

```
*NODE
0.0, 0.0,
<x1>, 0.0,
<x1>, <y1>,
0.0, <y1>,
*ELEMENT, TYPE=CPS4
1,2,4,3
```

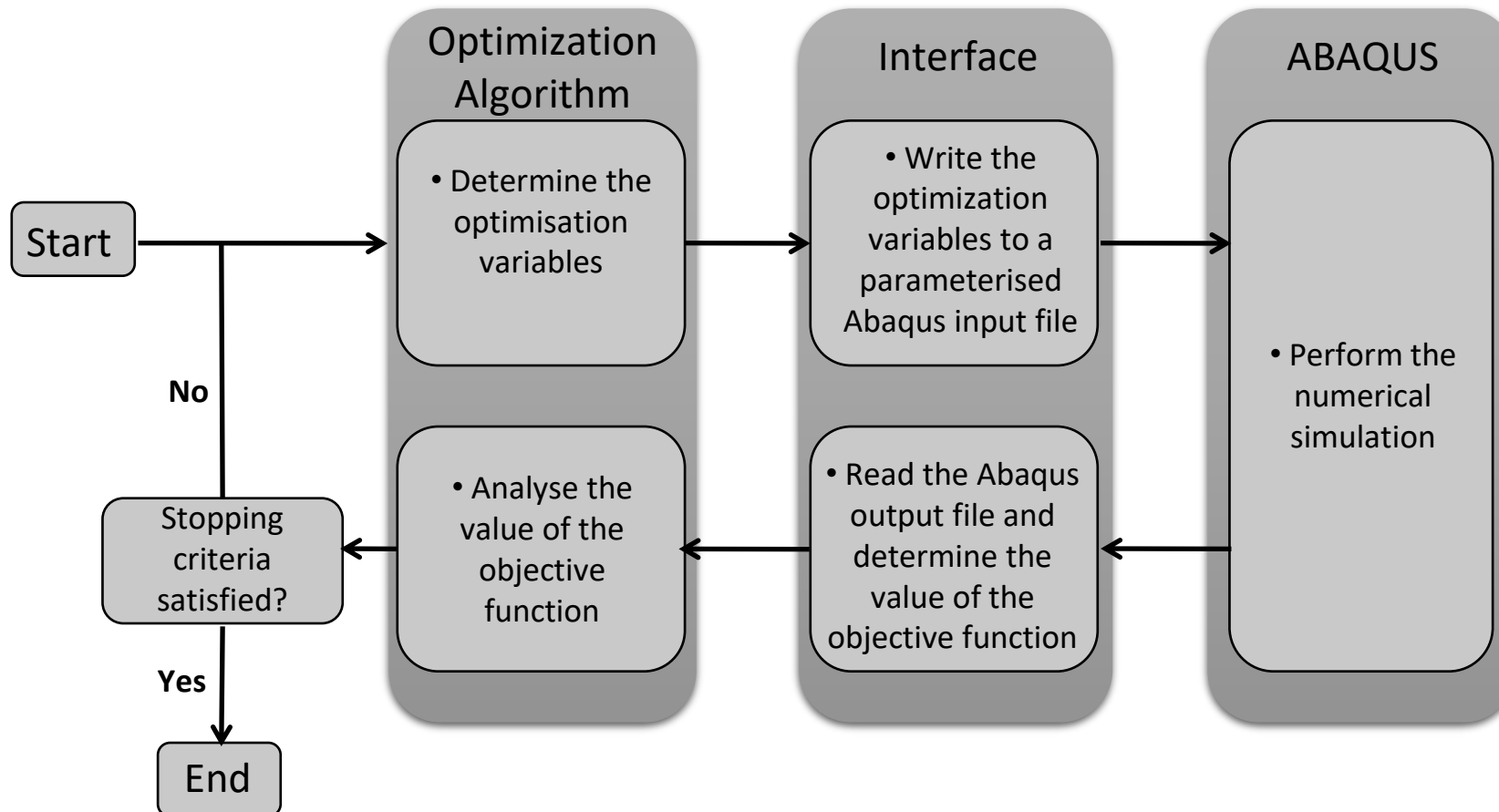
**Parameterised nodes**

The user can then use a **programming language** (Fortran, C++, etc.) to implement the optimisation algorithm and **link** it to Abaqus using the **parametric input file**



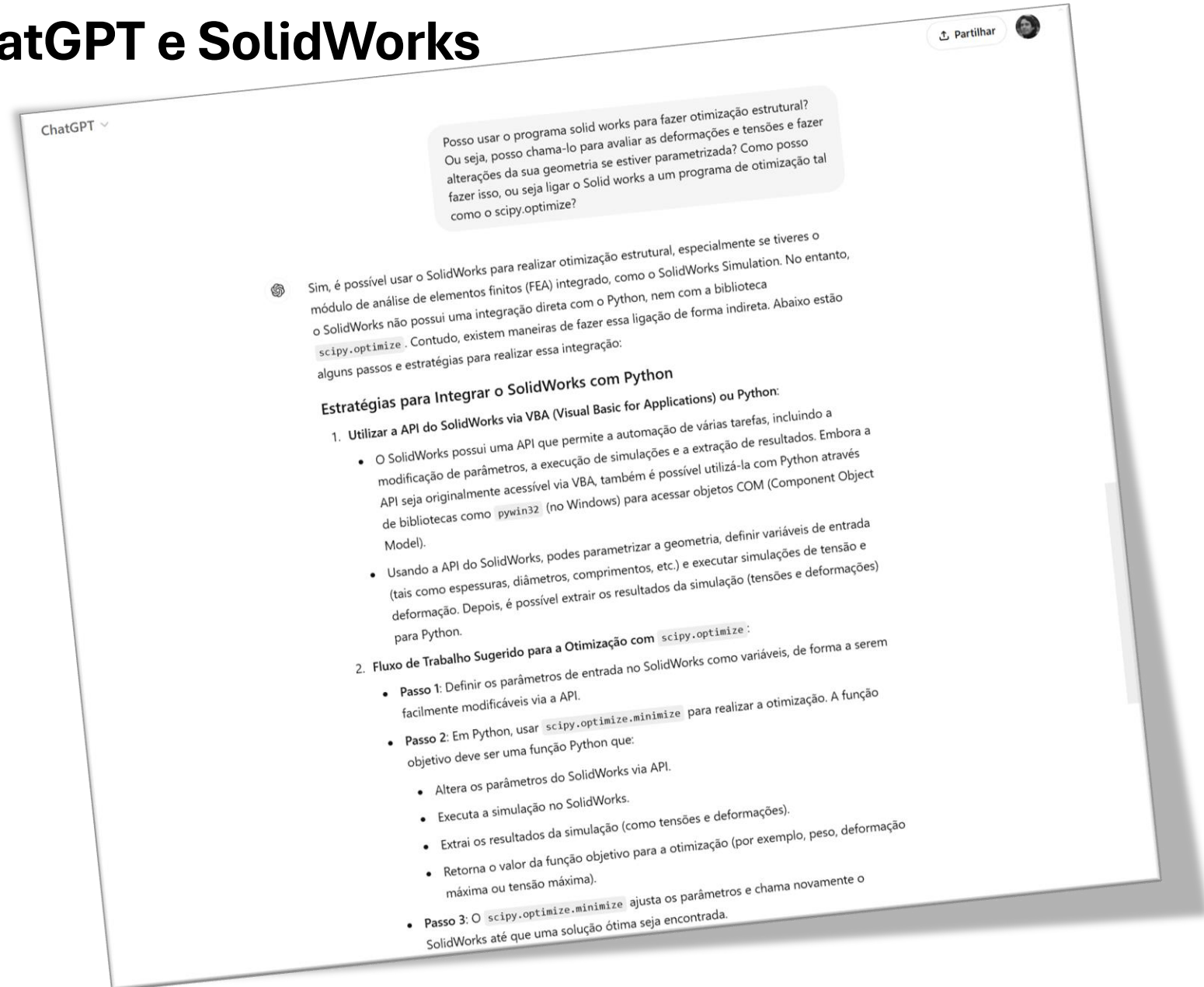
## Black-box problems: using FEA codes (e.g. Abaqus)

# Linking Optimization Algorithm - with FEA code Abaqus





# Otimização com Python, ChatGPT e SolidWorks





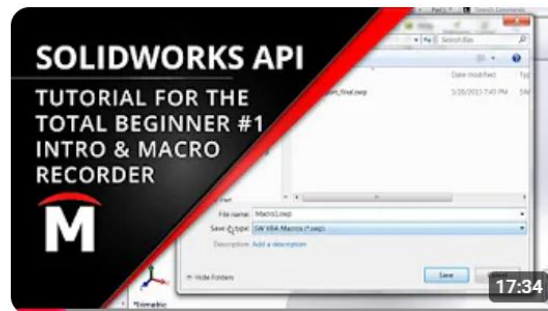
# Estratégias para Integrar o SolidWorks com Python

- **Utilizar a API do SolidWorks via VBA (Visual Basic for Applications) ou Python:**

- O SolidWorks possui uma API que permite a automação de várias tarefas, incluindo a modificação de parâmetros, a execução de simulações e a extração de resultados. Embora a API seja originalmente acessível via VBA, também é possível utilizá-la com Python através de bibliotecas como pywin32 (no Windows) para acessar objetos COM (Component Object Model).

- Usando a API do SolidWorks, podemos parametrizar a geometria, definir variáveis de entrada (tais como espessuras, diâmetros, comprimentos, etc.) e executar simulações de tensão e deformação. Depois, é possível extrair os resultados da simulação (tensões e deformações) para Python.

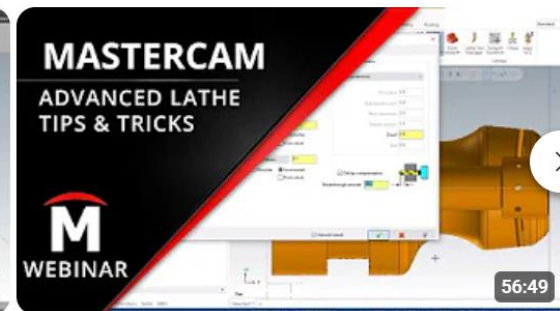
Ver <https://www.youtube.com/watch?v=UMT4v1pplmY>



SOLIDWORKS API for the Total Beginner – 1/6  
Introduction & Macro Recorder



Start to Finish Complex Machining Project -  
Mastercam 2024 Multi-Product Case Study



Mastercam Advanced Lathe Tips & Tricks Webinar  
18 mil visualizações • há 2 anos



# Otimização com Python e chatGPT

```
import win32com.client
from scipy.optimize import minimize

# Inicializar o SolidWorks
swApp = win32com.client.Dispatch("SldWorks.Application")
model = swApp.ActiveDoc # Obter o documento ativo no SolidWorks

def definir_parametros(R0, Ri):
    # Aqui definirias os parâmetros de entrada no SolidWorks
    # Exemplo: alterando variáveis de diâmetro e espessura
    parameter_R0 = model.Parameter("R0@Sketch1") # Nome do parâmetro
no SolidWorks
    parameter_Ri = model.Parameter("Ri@Sketch1")
    parameter_R0.SystemValue = R0 / 1000 # Conversão de mm para
metros
    parameter_Ri.SystemValue = Ri / 1000 # Conversão de mm para
metros
    model.EditRebuild3() # Atualizar o modelo após modificar os
parâmetros

def executar_simulacao():
    # Aqui executarias a simulação FEA no SolidWorks
    # Este é um pseudo-código; os detalhes dependem do setup da
simulação
    sim_manager = model.Extension.SimulationManager
    sim_manager.RunAnalysis() # Executa a análise
    # Esperar que a simulação termine antes de prosseguir
```

```
python Copiar código

import win32com.client
from scipy.optimize import minimize

# Inicializar o SolidWorks
swApp = win32com.client.Dispatch("SldWorks.Application")
model = swApp.ActiveDoc # Obter o documento ativo no SolidWorks

def definir_parametros(R0, Ri):
    # Aqui definirias os parâmetros de entrada no SolidWorks
    # Exemplo: alterando variáveis de diâmetro e espessura
    parameter_R0 = model.Parameter("R0@Sketch1") # Nome do parâmetro no SolidWorks
    parameter_Ri = model.Parameter("Ri@Sketch1")
    parameter_R0.SystemValue = R0 / 1000 # Conversão de mm para metros
    parameter_Ri.SystemValue = Ri / 1000 # Conversão de mm para metros
    model.EditRebuild3() # Atualizar o modelo após modificar os parâmetros

def executar_simulacao():
    # Aqui executarias a simulação FEA no SolidWorks
    # Este é um pseudo-código; os detalhes dependem do setup da simulação
    sim_manager = model.Extension.SimulationManager
    sim_manager.RunAnalysis() # Executa a análise
    # Esperar que a simulação termine antes de prosseguir

def obter_resultados():
    # Extrair resultados da simulação, como tensões ou deformações
    # Pseudo-código para obtenção de resultados
    max_tensao = model.SimulationResults.MaximumStress # Exemplo fictício
    return max_tensao

# Função objetivo para scipy.optimize
def objetivo(R):
    R0, Ri = R
    definir_parametros(R0, Ri)
    executar_simulacao()
    tensao = obter_resultados()
    return tensao # Objetivo: minimizar a tensão, por exemplo

# Limites das variáveis
limites = [(20, 100), (20, 100)]

# Estimativa inicial
estimativa_inicial = [50, 30]

# Executar a otimização
resultado = minimize(objetivo, estimativa_inicial, method='SLSQP', bounds=limites)

# Exibir o resultado
if resultado.success:
    print("Solução encontrada:")
    print("R0 =", resultado.x[0])
    print("Ri =", resultado.x[1])
    print("Valor mínimo da tensão =", resultado.fun)
else:
    print("Não foi possível encontrar uma solução.")
```





# Otimização com Python e chatGPT

```
def obter_resultados():
    # Extrair resultados da simulação, como tensões ou deformações
    # Pseudo-código para obtenção de resultados
    max_tensao = model.SimulationResults.MaximumStress # Exemplo fictício
    return max_tensao


# Função objetivo para scipy.optimize
def objetivo(R):
    R0, Ri = R
    definir_parametros(R0, Ri)
    executar_simulacao()
    tensao = obter_resultados()
    return tensao # Objetivo: minimizar a tensão, por exemplo

# Limites das variáveis
limites = [(20, 100), (20, 100)]

# Estimativa inicial
estimativa_inicial = [50, 30]

# Executar a otimização
resultado = minimize(objetivo, estimativa_inicial, method='SLSQP', bounds=limites)

# Exibir o resultado
if resultado.success:
    print("Solução encontrada:")
    print("R0 =", resultado.x[0])
    print("Ri =", resultado.x[1])
    print("Valor mínimo da tensão =", resultado.fun)
else:
    print("Não foi possível encontrar uma solução.")
    print("Mensagem de erro:", resultado.message)
```

python  Copiar código

```
import win32com.client
from scipy.optimize import minimize

# Inicializar o SolidWorks
swApp = win32com.client.Dispatch("SldWorks.Application")
model = swApp.ActiveDoc # Obter o documento ativo no SolidWorks

def definir_parametros(R0, Ri):
    # Aqui definirias os parâmetros de entrada no SolidWorks
    # Exemplo: alterando variáveis de diâmetro e espessura
    parameter_R0 = model.Parameter("R0@Sketch1") # Nome do parâmetro no SolidWorks
    parameter_Ri = model.Parameter("Ri@Sketch1")
    parameter_R0.SystemValue = R0 / 1000 # Conversão de mm para metros
    parameter_Ri.SystemValue = Ri / 1000 # Conversão de mm para metros
    model.EditRebuild3() # Atualizar o modelo após modificar os parâmetros

def executar_simulacao():
    # Aqui executarias a simulação FEA no SolidWorks
    # Este é um pseudo-código; os detalhes dependem do setup da simulação
    sim_manager = model.Extension.SimulationManager
    sim_manager.RunAnalysis() # Executa a análise
    # Esperar que a simulação termine antes de prosseguir

def obter_resultados():
    # Extrair resultados da simulação, como tensões ou deformações
    # Pseudo-código para obtenção de resultados
    max_tensao = model.SimulationResults.MaximumStress # Exemplo fictício
    return max_tensao

# Função objetivo para scipy.optimize
def objetivo(R):
    R0, Ri = R
    definir_parametros(R0, Ri)
    executar_simulacao()
    tensao = obter_resultados()
    return tensao # Objetivo: minimizar a tensão, por exemplo

# Limites das variáveis
limites = [(20, 100), (20, 100)]

# Estimativa inicial
estimativa_inicial = [50, 30]

# Executar a otimização
resultado = minimize(objetivo, estimativa_inicial, method='SLSQP', bounds=limites)

# Exibir o resultado
if resultado.success:
    print("Solução encontrada:")
    print("R0 =", resultado.x[0])
    print("Ri =", resultado.x[1])
    print("Valor mínimo da tensão =", resultado.fun)
else:
    print("Não foi possível encontrar uma solução.")
```

Novembro 2024

