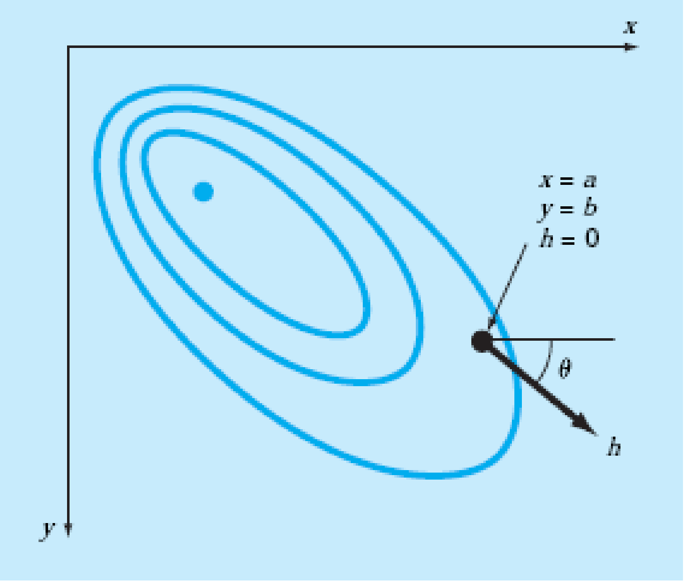
Steepest Ascent (Hill Climb) for Optimisation

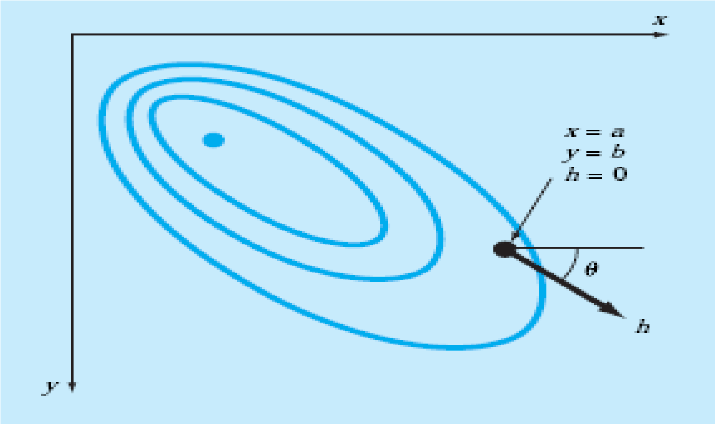
1. There are a number of multivariate optimisation techniques commonly in use for unconstrained multivariate optimisation.
2. **We are going to focus on gradient methods in this course**, which are among the most efficient and most intuitive to use.
3. The solution to a multivariate problem is equivalent to **finding the shortest route to the top of a high mountain** which has a variety of local heights, valleys and ridges randomly scattered across its surface.

A multivariate optimisation involving the solution of a surface function f(x,y) for a maximum or minimum value



Steepest Ascent (Hill Climb) for Optimisation

A multivariate optimisation involving the solution of a surface function f(x,y) for a maximum or minimum value



4. The first step in solving such a system is to identify where you are on the surface (e.g. the point [x,y] = [a,b] above).
5. The curve of travel ***in any direction*** from [a,b] is expressible by a function g(h), where h is set to zero at [a,b]. **g(h) is the optimisation path function**.
6. At the peak of this curve of travel, g'(h) = 0, the derivative function (below) is zero. The value of h at this point is used to calculate the next point (x,y).

$$g'(h) = \frac{\partial f}{\partial x}\cos\theta + \frac{\partial f}{\partial y}\sin\theta = 0$$

Steepest Ascent (Hill Climb) for Optimisation

7. The path of Steepest Ascent is shown in the following figure

$$x = x_0 + \frac{\partial f}{\partial x} h$$

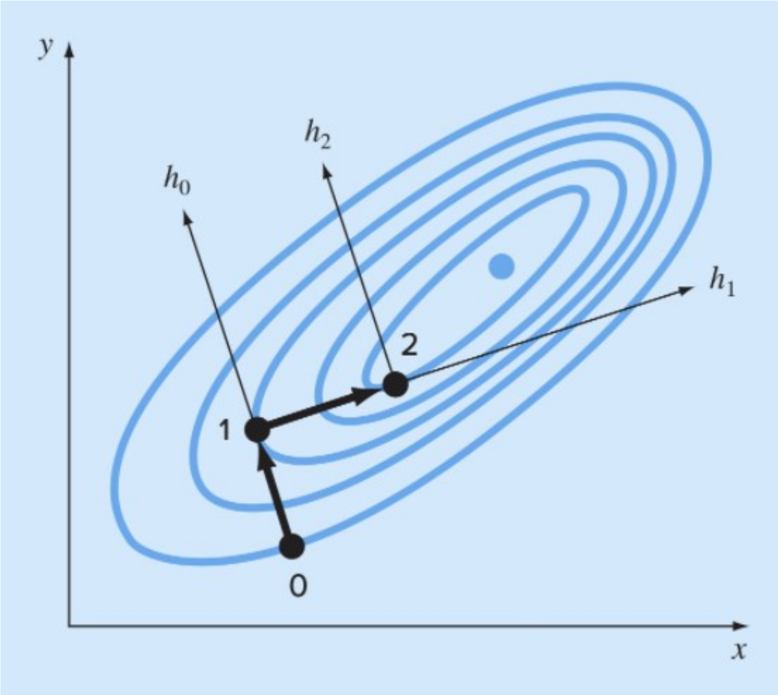
(A)

$$y = y_0 + \frac{\partial f}{\partial y} h$$

$$x = 1 + 3h$$

(B)

$$y = 2 + 4h$$



8. The next point (x, y) at each iteration from an initial point (x_0, y_0) is given by the general equations above.

9. Thus, in this example, any point lying on the axis h (the path) has the general equation (B) if the initial point (x_0, y_0) is (1,2) and $\nabla f = 4\mathbf{i} + 8\mathbf{j}$

Steepest Ascent (Hill Climb) for Optimisation

10. The following example describes how Steepest Ascent is implemented.
11. Consider the following equation:

$$f(x, y) = 2xy + 2x - x^2 - 2y^2$$

12. First, calculate the partial derivatives for this equation:

$$\frac{\partial f}{\partial x} = 2y + 2 - 2x = 2(1) + 2 - 2(-1) = 6$$
$$\frac{\partial f}{\partial y} = 2x - 4y = 2(-1) - 4(1) = -6$$

13. This gives the gradient vector to be:

$$\nabla f = 6\mathbf{i} - 6\mathbf{j}$$

Steepest Ascent (Hill Climb) for Optimisation

14. The implementation of the technique requires an equation in h (the path of the optimisation along the surface) to be written:

$$f\left(x_0 + \frac{\partial f}{\partial x}h, y_0 + \frac{\partial f}{\partial y}h\right) = f(-1+6h, 1-6h)$$
$$= 2(-1+6h)(1-6h) + 2(-1+6h) - (-1+6h)^2 - 2(1-6h)^2$$

15. The partial derivatives in this expression are evaluated at the initial point (x = -1, and y = 1).

16. When the above expression is evaluated fully and simplified we get:

$$g(h) = -180h^2 + 72h - 7$$

17. Solution of the derivative of this equation for g'(h) = 0 will give us h that gives the maximum value of the g(h) elevation function. Determining h for g'(0) = 0 is equivalent to 1-dimensional optimisation. A simplified challenge!

18. Then back-substitute this value of h into $x = x_0 + \frac{\partial f}{\partial x}h$ $\quad y = y_0 + \frac{\partial f}{\partial y}h$

Code for Steepest Ascent (the g-path)

Initialise the programme importing all necessary packages.

```
def HOpt(F,dFx,dFy,x,y):
    import sympy as sym
    from sympy import symbols, solve
    import matplotlib.pyplot as plt
```

Create a symbolic variable for h, so that it can be solved for.

```
    hsym = symbols('hsym')
```

Create a series of empty arrays to store successive values

```
    xlist = []
    ylist = []
    flist = []
    dfxlist = []
    dfylist = []
```

Set up a loop that solves h and updates x and y

```
    for i in range(0,10,1):
        xold = x
        yold = y

        dfx = dFx(x)
        dfy = dFy(y)
```

Define the g function and its arguments)

```
g = F(x+dfx*hsym, y+dfy*hsym)
    hexpr = sym.diff(g, hsym)
```

Solve for g'(0) to find h at the peak of the path

```
    hsolved = solve(hexpr)
    hopt = hsolved[0]
```

hopt is a numeric value from the hsolved array

```
    x = xold + hopt*dfx
    y = yold + hopt*dfy
```

Update for the new point along the hill.

```
    Fxy = F(x,y)
    dfx = dFx(x)
    dfy = dFy(y)
```

Update the function value and derivatives at this new point along path g.

Code for Steepest Ascent (the g-path)

Append the latest values of variables to the arrays created earlier. This allows a plot of the path to be made.

```
        xlist.append(x)
        ylist.append(y)
        flist.append(Fxy)
        dfxlist.append(dfx)
        dfylist.append(dfy)
```

If the derivatives of both functions are very low, accept the result as the peak of the hill.

```
if dfx <= 0.0001 and dfy <=
0.0001:
            break
```

Print the final values of the variables at the peak (x,y).

```
    print(x,y,Fxy,dfx,dfy)
```

Plot x against the hill height, f(x,y).

```
    plt.plot(xlist,flist,'o')
    plt.show()
```

Plot x against y along the optimisation path, f(x,y).

```
    plt.plot(xlist,ylist,'o')
    plt.show()
```

Define the function and its derivatives.

```
def F(x,y):
    return 2*x*y+2*x-x**2-
2*y**2

def dFx(x):
    return 2*y+2-2*x

def dFy(y):
    return 2*x-4*y
```

Standard Calling Commands

```
x = -1.
y = 1.
print(HOpt(F,dFx,dFy,x,y))
```

Computational Methods and Modelling

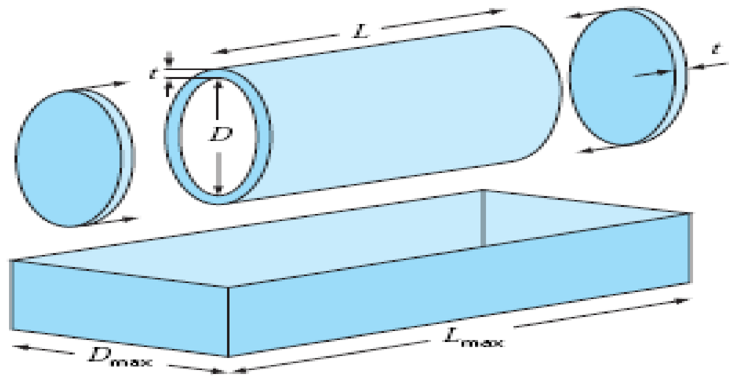
Lecture 11: Dr. Edward McCarthy

Topic 2: Optimisation Example: Least Cost Design of a Tank

THE UNIVERSITY of EDINBURGH
School of Engineering

Constrained Non-Linear Optimisation: Finding Minima and Maxima

1. Let's look at designing a tank with least cost as a practical example of optimisation. (Chapra p. 413).
2. You are asked to design a cylindrical tank as shown with geometrical variables that have not yet been fully optimised (you will do this).



Parameter	Symbol	Value	Units
Required volume	V_o	0.8	m^3
Thickness	t	3	cm
Density	ρ	8000	kg/m^3
Bed length	L_{max}	2	m
Bed width	D_{max}	1	m
Material cost	c_m	4.5	\$/kg
Welding cost	c_w	20	\$/m

Optimisation: Finding Minima and Maxima

3. The **volume of the tank is fixed** at 0.8 m^3, and **the tank thickness is defined** by waste handling regulations (3 cm).
4. Your constraints are that the tank must be transportable on a truck bed with **fixed, pre-defined dimensions (2 x 1 m)**
5. Note that these represent maximum value constraints; you may design a tank with l and d lower than these maxima.
6. The tank expense consists of 1) material expense (steel), and 2) welding expense, which is proportional to the length of welding seams in the tank.
7. **Solution:** The first step is to **define an objective function for cost** that you will minimise via an optimisation routine.

$$C = c_m m + c_w \ell_w$$

8. Here, C is total cost, c_m is material cost per unit mass of material, c_w is welding cost per m of weld, and l_w is weld length.

THE UNIVERSITY of EDINBURGH
School of Engineering

Optimisation: Finding Minima and Maxima

9. Now we need to define the volume of the cylinder and understand the contributions of cylinder length and diameter to its volume.
10. The volume of its cylindrical side wall can be computed as follows

$$V_{\text{cylinder}} = L\pi\left[\left(\frac{D}{2}+t\right)^2 - \left(\frac{D}{2}\right)^2\right]$$

11. The volume of each of its end plates is:

$$V_{\text{plate}} = \pi\left(\frac{D}{2}+t\right)^2 t$$

12. Thus, the mass of the total tank is given by:

$$m = \rho\left\{L\pi\left[\left(\frac{D}{2}+t\right)^2 - \left(\frac{D}{2}\right)^2\right] + 2\pi\left(\frac{D}{2}+t\right)^2 t\right\}$$

13. Here, we have multiplied the tank material volume by the steel density, rho.
11. Weld length is defined by: $\ell_w = 2\left[2\pi\left(\frac{D}{2}+t\right) + 2\pi\frac{D}{2}\right] = 4\pi(D+t)$

Optimisation: Finding Minima and Maxima

15. Now we calculate the constraints on the problem. Firstly, the volume, V_0, of the tank (i.e., its hollow volume) is fixed.

$$V = \frac{\pi D^2}{4} L$$

16. The remaining constraints are the bed dimensions of the truck as follows:

$$L \leq L_{\max}$$
$$D \leq D_{\max}$$

15. Our task is now expressible as follows (mathematically), where 4.5 and 20 are the cost coefficients of the material and the welding, respectively Minimise $C = 4.5m + 20\ell_w$

15. Subject to the following conditions:

$$\frac{\pi D^2 L}{4} = 0.8$$
$$L \leq 2$$
$$D \leq 1$$

THE UNIVERSITY *of* EDINBURGH
School of Engineering

Optimisation: Finding Minima and Maxima

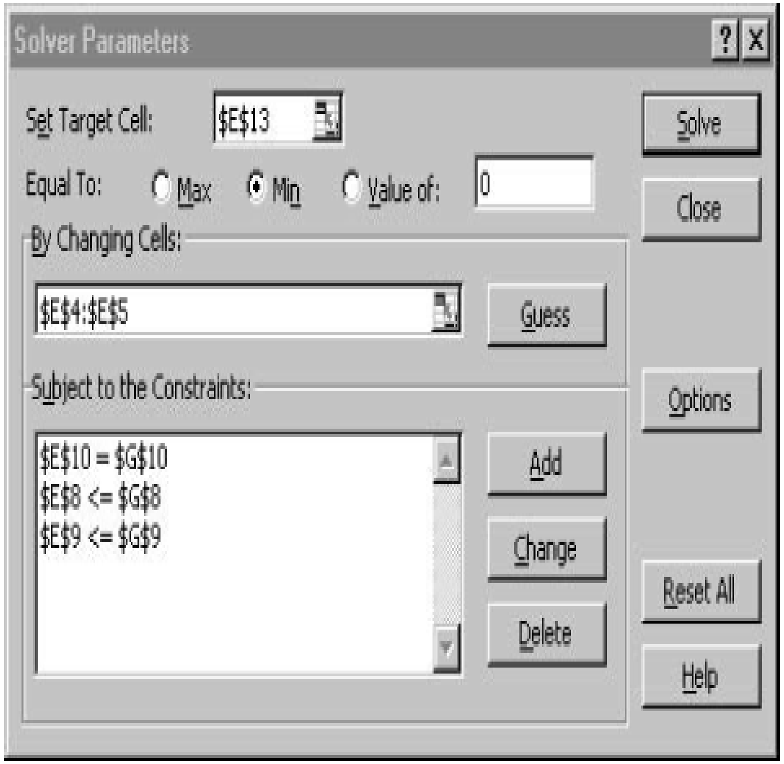
1. The above problem is an example of constrained non-linear optimisation
2. To solve this we have a number of options.
3. One of them is **to use the in-built Solver of Microsoft Excel** which can handle a multivariate optimisation problem (which this is).
4. In Excel, we will enter all input variables in one set of cells, and the formulae for the outputs (mass and weld length) in others. (next slide)
5. We will also define the constraints on the optimisations explicitly into the Solver dialogue box. (next slide).
6. You will notice that Solver can handle root-finding problems, and optimisation problems formulated to result in a minimum or maximum function result.
7. No explicit coding is required: this is embedded in the application.

Optimisation: Finding Minima and Maxima

Inputs and formulae for this problem in an Excel spreadsheet

	A	B	C	D	E	F	G
1	**Optimum tank design**						
2							
3	**Parameters:**			**Design variables:**			
4	V0	0.8		D	1		
5	t	0.03		L	2		
6	rho	8000					
7	Lmax	2		**Constraints:**			
8	Dmax	1		D	**1**	<=	1
9	cm	4.5		L	**2**	<=	2
10	cw	20		Vol	**1.570796**	=	0.8
11							
12	**Computed values:**			**Objective function:**			
13	m	1976.791		C	9154.425		
14	lw	12.94336					
15							
16	Vshell	**0.19415**					
17	Vends	**0.052948**					

Solver Panel where we enter target, cells to be varied (inputs) and constraints.



Computational Methods and Modelling

Lecture 11: Dr. Edward McCarthy

Topic 1: Optimisation Example: Least Cost Design of a Tank

THE UNIVERSITY *of* EDINBURGH
School of Engineering

Optimisation: Finding Minima and Maxima

1. Now, the result of the optimisation appears in the spreadsheet itself (right panel) where all values have been updated in line with the specifications made in the Solver dialogue box before it ran. The original sheet is on the left below: the new sheet is on the right.
2. Here, we see that the mass of the tank has been reduced from 1976 kg to 1215 kg, while the weld seam has only been reduced slightly in length from 12.94 m to 12.73 m.
3. The cost has dropped significantly! (See Cell E13 in both panels). This is an effective way of solving many such constrained multivariate problems relevant to engineering.

	A	B	C	D	E	F	G
1	Optimum tank design						
2							
3	Parameters:			Design variables:		INPUT	
4	V0	0.8		D	1	VALUES	
5	t	0.03		L	2		
6	rho	8000					
7	Lmax	2		Constraints:			
8	Dmax	1		D	1	<=	1
9	cm	4.5		L	2	<=	2
10	cw	20		Vol	1.570796	=	0.8
11							
12	Computed values:			Objective function:			
13	m	1976.791		C	9154.425		
14	lw	12.94336					
15							
16	Vshell	0.19415					
17	Vends	0.052948					

	A	B	C	D	E	F	G
1	Optimum tank design						
2							
3	Parameters:			Design variables:		OUTPUT	
4	V0	0.8		D	0.98351	VALUES	
5	t	0.03		L	1.053033		
6	rho	8000					
7	Lmax	2		Constraints:			
8	Dmax	1		D	0.98351	<=	1
9	cm	4.5		L	1.053033	<=	2
10	cw	20		Vol	0.799999	=	0.8
11							
12	Computed values:			Objective function:			
13	m	1215.206		C	5723.149		
14	lw	12.73614					
15							
16	Vshell	0.100587					
17	Vends	0.051314					

Computational Methods and Modelling

Lecture 11: Dr. Edward McCarthy

Topic 1: Optimisation Examples

Example 1: Lagrange Multiplier Technique for Multivariate Constrained Optimisation

1. In this lecture, we look at the use of the Lagrange multiplier to re-express a constrained multivariate optimisation problem where the constraint must be an equality constraint.

2. The use of a Lagrange multiplier solves it more efficiently ***by incorporating the constraint function into the objective function.***

3. A compound expression for the objective and constrain functions gives the following:

$$L(y) = f(x) + \gamma g(x)$$

4. Here, f(x) is the objective function and g(x) is the equality constraint.

5. The next example shows how this method works in practice.

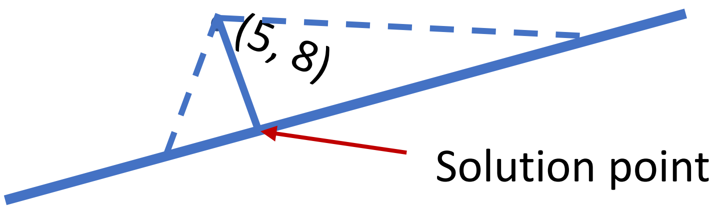
Computational Methods and Modelling

Lecture 11: Dr. Edward McCarthy

Topic 1: Optimisation Examples

Example 1: Lagrange Multiplier Technique for Multivariate Constrained Optimisation

$$F^*(x, y, \lambda) = (x-5)^2 + (y-8)^2 + \lambda(xy-5)$$



- Determine the smallest distance from the point (5, 8) to an unknown point (x,y) on the line *xy* = 5 (which forms an equality constraint condition). The two dotted lines are incorrect estimates for (x,y).

- You will notice that the objective function for this problem is the equation of a circle with its centre at (5,8). Where the circle touches the line, that point (x,y) will be the solution to our optimisation.

- The derivative of the circle function gives its tangent. At the shortest approach of (5,8) to the function (x,y), y = 5/x, the tangent will be equal to this line.

Computational Methods and Modelling

Lecture 11: Dr. Edward McCarthy

Topic 1: Optimisation Examples

Example 1: Lagrange Multiplier Technique for Multivariate Constrained Optimisation

1. In this example, the expression for the compound equation contains the objective function plus the equality constraint in brackets multiplied by a constant λ, (the Lagrange multiplier) that needs to be calculated to solve the problem.

$$F^*(x, y, \lambda) = (x-5)^2 + (y-8)^2 + \lambda(xy-5)$$

2. The optimum point will be found when the components of the main objective equation are differentiated and the differentiated expressions are solved for zero. In this case, the relevant equations of the system are:

$$\frac{\partial F^*}{\partial x} = 2(x-5) + \lambda y = 0$$

$$\frac{\partial F^*}{\partial y} = 2(y-8) + \lambda x = 0$$

$$g(x) = xy - 5 = 0$$

Computational Methods and Modelling

Lecture 11: Dr. Edward McCarthy

Topic 1: Optimisation Examples

Example 1: Lagrange Multiplier Technique for Multivariate Constrained Optimisation

1. The three equations define the identity of a common optimum point that satisfies each, therefore **we can solve for the optimum point by solving all three equations as a system of simultaneous equations.**

2. However, we could not use Gauss Elimination in this case, because g(x) uniquely contains a term in xy, whereas the other two equations do not. That is, **it is not a system of linear equations.**

3. In this case, the Lagrange Multiplier technique can be used: (next page)

4. When the algorithm is finished in this case, the result of the optimisation gives the co-ordinate values of the point required:

$$\mathbf{X = [0.6556\ (x)\ 7.6265\ (y)\ 1.1393\ (\lambda)]^T}$$

Computational Methods and Modelling

Lecture 11: Dr. Edward McCarthy

Topic 1: Optimisation Examples

Example 1: Lagrange Multiplier Technique for Multivariate Constrained Optimisation

```
import numpy as np
from scipy.optimize import minimize

def objective(X):
    x, y = X
    return (x-5)**2 + (y-8)**2

#This is the constraint function that
has lambda as a coefficient.
def eq(X):
    x, y = X
    return x*y - 5

import autograd.numpy as np
from autograd import grad

def F(L):
    'Augmented Lagrange function'
    x, y, _lambda = L
    return objective([x, y]) +
_lambda * eq([x, y])
```

```
# Gradients of the Lagrange
function

dfdL = grad(F, 0)

# Find L that returns all zeros in
this function.

def obj(L):
    x, y, _lambda = L
    dFdx, dFdy, dFdlam = dfdL(L)
    return [dFdx, dFdy, eq([x,
y])]
from scipy.optimize import fsolve
x, y, _lam = fsolve(obj, [0.0,
0.0, 0.0])
print(f'The answer is at {x, y}')
```

Computational Methods and Modelling

Lecture 11: Dr. Edward McCarthy

Topic 1: Optimisation Examples

Example 2: Maximise the Revenue of an Engineering Project

1. The revenue of an engineering project can be expressed by the following equation, where s is the mass of steel used and h is the number of hours of labour employed on the project:

$$R(h, s) = 160\ h^{2/3} s^{1/3}$$

2. The project has a limited budget of £20,000, and the price of steel is £0.15 per kg, while the labour rate is: £20 per hour.
3. Use an appropriate optimisation technique to maximise revenue while maintaining costs within the allowable budget.

Solution 2: Maximise the Revenue of an Engineering Project

1. It is clear that the objective equation is the one given to calculate revenue, and that we need to maximise the value of R while not allowing total cost to exceed £20,000

$$R(h, s) = 160\ h^{2/3} s^{1/3}$$

2. We now need to express what the constraint is mathematically. It is clear that we need an equation for total cost in terms of the individual costs of steel and labour as follows:

$$20h + 0.15s = 20{,}000$$

3. This is an equality constraint. We could choose to aim for a budget below £20,000 to save cost, but it makes more sense to maximise revenue by using up all of our budget. This means we have an equality constraint rather than an inequality constraint (less than).

4. To solve this we can use the Langrange Multiplier Technique.

Solution 2: Maximise the Revenue of an Engineering Project

1. To do this we need to combine the objective function and the equality constraint into a combined equation as follows:

$$L(y) = f(x) + \gamma g(x)$$

2. This allows us to write for this case that:

$$R(h,s) = 160\ h^{2/3} s^{1/3} - \lambda(20h + 0.15h - 20{,}000)$$

3. Now that we have the form of the equation we can use the code we previously used to solve the problem as per the next slide.

Computational Methods and Modelling

Lecture 11: Dr. Edward McCarthy

Topic 1: Optimisation Examples

Solution 2: Maximise the Revenue of an Engineering Project

```
def objective(X):
    x, y = X
    return (160*x**0.66)*(y**0.33)

#This is the constraint function that
has lambda as a coefficient.
def eq(X):
    x, y = X
    return 20*x + 0.15*y - 20000.

import autograd.numpy as np
from autograd import grad

def F(L):
    'Augmented Lagrange function'
    x, y, _lambda = L
    return objective([x, y]) + _lambda *
eq([x, y])

# Gradients of the Lagrange function
dfdL = grad(F, 0)
```

```
# Find L that returns all zeros in
this function.
def obj(L):
    x, y, _lambda = L
    dFdx, dFdy, dFdlam = dfdL(L)
    return [dFdx, dFdy, eq([x, y])]

from scipy.optimize import fsolve
x, y, _lam = fsolve(obj, [1., 1.,
1.0])
print(f'The answer is at {x, y}')
```

Answer:

666.66 hours of labour,

44.44 tonnes of steel