Lecture 4

Basics of optimisation theory
Outline

Optimality conditions
   Necessary conditions
   Sufficient conditions

Optimisation methods
   Analytical
   Iterative methods
   - Direct methods
   - Indirect methods
   - Direction based
   - Function comparison methods
Constrained and unconstrained optimisation

Unconstrained

\[ \min_{x_1, \ldots, x_n} f(x_1, x_2, \ldots, x_n) \]

- majority of regression and approximation problems

seldom arises in practical applications

Constrained

\[ \min_{x_1, \ldots, x_n} f(x_1, x_2, \ldots, x_n) \]

Subject to

\[ h(x_1, x_2, \ldots, x_n) = 0 \]

\[ g(x_1, x_2, \ldots, x_n) \leq 0 \]

Optimality conditions are essentially identical
Unconstrained example

Fit

\[ y = a \cdot x_1^{b_1} \cdot x_2^{b_2} \]

Into set of data:

<table>
<thead>
<tr>
<th>( Y_{\text{exp,}i} )</th>
<th>( x_{1,i} )</th>
<th>( x_{2,i} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>46.5</td>
<td>2.0</td>
<td>7.1</td>
</tr>
<tr>
<td>57.1</td>
<td>6.0</td>
<td>6.8</td>
</tr>
<tr>
<td>128.3</td>
<td>9.0</td>
<td>6.4</td>
</tr>
<tr>
<td>72.8</td>
<td>11.0</td>
<td>3.5</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>151.3</td>
<td>12.7</td>
<td>6.7</td>
</tr>
</tbody>
</table>

\[ \min_{a,b_1,b_2} \sum_{i=1}^{n} (y_{\text{exp,}i} - y_{\text{pred,}i})^2 = \]

\[ = \sum_{i=1}^{n} (y_{\text{exp,}i} - a \cdot x_{1,i}^{b_1} \cdot x_{2,i}^{b_2})^2 \]

DOF?
Necessary conditions

If \((x_1^*, x_2^*, \ldots x_n^*)\) is an optimum for
\[
\min_{x_1, \ldots x_n} f(x_1, x_2, \ldots x_n)
\]
then there should be
\[
\nabla f(x^*) = 0
\]
or
\[
\left(\frac{\partial f}{\partial x_1}\right)_{x_1^*} = 0
\]
\[
\ldots
\]
\[
\left(\frac{\partial f}{\partial x_n}\right)_{x_n^*} = 0
\]

A point satisfying \(\ominus\) is called a stationary point
Example

Minimise

\[ f(x_1, x_2) = x_1^4 + 2x_1 x_2 \]

Necessary conditions

\[
\begin{align*}
(\frac{\partial f}{\partial x_1}) &= 0 \\
(\frac{\partial f}{\partial x_2}) &= 0
\end{align*}
\]

\[
\begin{align*}
4x_1^3 + 2x_2 &= 0 \\
2x_1 &= 0
\end{align*}
\]

\[ x_1^* = x_2^* = 0 \]

But is it an maximum or a minimum?
The Hessian matrix

\[ F(x_1, x_2, \ldots x_n) \]

\[
\begin{pmatrix}
\frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\
\vdots & \ddots & \vdots \\
\frac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_n^2}
\end{pmatrix}
\]

The Hessian is always a **square** matrix
Positive and negative definite matrices

A square matrix

\[ A = \begin{pmatrix} a_{ii} & \ldots & a_{in} \\ \vdots & \ddots & \vdots \\ a_{ni} & \ldots & a_{nn} \end{pmatrix} \]

with

- all eigenvalues positive \( \Rightarrow \) positive definite
- all eigenvalues negative \( \Rightarrow \) negative definite
- eigenvalues positive and negative \( \Rightarrow \) indefinite
Sufficient conditions

If \((x_1^*, x_2^*, \ldots x_n^*)\) that satisfied necessary conditions and yields

- a positive definite \(H(x_1^*, \ldots x_n^*)\) is minimum
- a negative definite \(H(x_1^*, \ldots x_n^*)\) is maximum
- an indefinite \(H(x_1^*, \ldots x_n^*)\) is a saddle point

Use

necessary conditions \(\Rightarrow\) stationary points
sufficient conditions \(\Rightarrow\) maxima, minima, saddle points
Illustration

\[ f(x_1, x_2) = x_1^2 - x_1 x_2 + x_2^2 - 3x_2 \]

Find all extrema

(1) Necessary conditions
\[ \begin{align*}
\frac{\partial f}{\partial x_1} &= 0 = 2x_1 - x_2 \\
\frac{\partial f}{\partial x_2} &= 0 = -x_1 + 2x_2 - 3
\end{align*} \]
\[ \Rightarrow (x_1^*, x_2^*) = (1, 2) \]

(2) Sufficient conditions
\[ H(x_1, x_2) = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix} \]

Eigenvalues: of \( H \)
\[ \begin{vmatrix} 2-\lambda & -1 \\ -1 & 2-\lambda \end{vmatrix} = 0 \Rightarrow \begin{cases} \lambda_1 = 1 \\ \lambda_2 = 3 \end{cases} \]

Then: \((x_1^*, x_2^*)\) is a global minimum
Analytical methods

Basic idea

Use necessary and sufficient conditions to determine optima. Analytical expressions are required for
- first order derivatives
- second order derivatives
- expressions for necessary conditions

Difficulties

- Conditions yield nonlinear sets of equations that are difficult (or impossible) to solve; multiple solutions
- Analytical expressions for derivatives may be difficult to calculate
Iterative methods

Direct methods
No use of derivatives
- smooth functions
- non-smooth functions

Indirect methods
Use of derivative information
- first order
- second order

Function methods
- Generate sequence of points
- Compare and screen inferior points
- Iterate

Direction based methods
- Generate directions of improvement
- Optimise against selected direction
- Iterate
Univariate search
(direction based method)

Steps: move along $x_1$ and $x_2$
1 - Select initial point $x^0$
2 - Optimise along $x_1 \rightarrow x^1$
3 - Optimise along $x_2 \rightarrow x^2$
4 - Iterate from (2)

Advantages: very simple
Disadvantages: very slow
(especially at later iterations)
Powell’s method

Basic idea: Employ univariate search to identify characteristic direction(s). Use the characteristic direction in the remaining stages.

Steps
- Univariate search $x^0 \rightarrow x^1 \rightarrow x^2$
- Develop direction $d$
- Optimise along $d$: $x^2 \rightarrow x^*$
Indirect methods

Basic idea: Why use arbitrary directions when important directions are available from derivatives?

Steepest Descent:

Direction $d = - \nabla f(x^*)$

Method
1 - Guess $x^0$
2 - Calculate search direction: $d^i = - \nabla f(x^i)$
3 - Find step site that minimises $f(x)$
4 - Set $x^{i+1} = x^i + a \cdot d$
5 - $i = i+1$ and iterate from (2)
Newton’s method: second order information

For nonlinear systems a better approximation for the improving direction is

\[ d^i = -\left\{H(x^i)\right\}^{-1} \nabla f(x^i) \]

– For quadratic systems \((H(x^i) = \text{constant})\) the method reduces to steepest descent

– Search direction requires inversion of a matrix, a time consuming and expensive process

– Method has fastest convergence when close to the optimum

– Method is good only for definite Hessians
Generalisation of iterative methods

Steepest descent: \[ d^i = -\nabla f(x^i) \]

Newton method: \[ d^i = -\left\{ H(x^i) \right\}^{-1} \cdot \nabla f(x^i) \]

\[ d^i = -\left\{ B^i \right\}^{-1} \nabla f(x^i) \]

A positive definite matrix

**BFGS**
Use an approximate version of the Hessian that does not require inversion, but only calculation of a new row and column at each iteration

**Marquandt method**
\[ B^i = \left\{ H^i \right\} + \lambda I \]
Select \( \lambda \) to make \( B^i \) a positive matrix
Conjugate gradient methods

Conjugate directions with respect to $Q$

$$d_1^T Q d_2 = 0$$

Selection of optimisation points

$$x^{i+1} = x^i + a_k \cdot \nabla f(x^i)$$

$$a_k = -\frac{\nabla^T f(x^i) d_i}{(d^i)^T H(x^i) d_i}$$

- Develop conjugate directions around the Hessian matrix

- n-dimension quadratic problem: Guarantee convergence within n iterations
Successive Quadratic Programming

**Basic idea** Use quadratic functions to approximate Hessian

Use the assumption on the quadratic approximation to

- Calculate the intermediate optima
- Update approximation at each iteration
- Idea extends SLP approach to an additional dimension

The SQP is used as a core idea in several solvers e.g.

- CONOPT, SNOPT
- RT-OPT (Aspen Plus)
Function comparison methods

Useful for nonsmooth functions or when derivative information is unreliable or difficult to achieve (i.e. discontinuous functions, functions with unbounded derivatives etc.)

Region elimination methods
Systematically reduce feasible region by discounting the portions that do not contain the optimum

Polytope methods
(Also referred to as *simplex*) Generate polytope and update its corner points towards the optimum solution
Polytope method

1 Calculate $f$ at corner points
2 Calculate $f$ at the centroid
3 Reflect worse point to $A$ centroid
4 Develop new polytope and iterate
Polytope method: illustration
Constrained optimisation

So far, discussion assumes unconstrained optimisation problems,
but what about constrained optimisation?

Lagrange function
Lagrange function

Solution of … … is identical to

\[
\min_{x} f(x)
\]
Subject to
\[
\begin{align*}
  h(x) &= 0 \\
  g(x) &\leq 0
\end{align*}
\]
Lagrangian function (one inequality and one equality):

\[
L(x, \lambda, \mu) = f(x) + \lambda \ h(x) + \mu \ g(x)
\]

Or, more generally

\[
L(x, \lambda, \mu) = f(x) + \sum_{i=1}^{n} \lambda_i h_i(x) + \sum_{i=1}^{r} \mu_i g_i(x)
\]

\[
\mu_i \geq 0
\]
Illustration

\[ \min 4x_1^2 + 5x_2^2 \]

Subject to

\[ h(x_1, x_2) = 2x_1 + 3x_2 - 6 = 0 \]

Lagrangian function

\[ L(x_1, x_2, \lambda) = 4x_1^2 + 5x_2^2 + \lambda (2x_1 + 3x_2 - 6) \]

Optimality conditions

\[
\begin{align*}
\frac{\partial L}{\partial x_1} &= 0 = 8x_1 + 2\lambda & x_1 &= 1.071 \\
\frac{\partial L}{\partial x_2} &= 0 = 10x_2 + 3\lambda & x_2 &= 1.1286 \\
\frac{\partial L}{\partial \lambda} &= 0 = 2x_1 + 3x_2 - 6 & \lambda &= -30/7
\end{align*}
\]

⇒ Significant similarities between constrained and unconstrained optimisation
Optimisation methods for unconstrained optimisation

General method
(i) start from initial point $x^0$
(ii) determine direction of improvement $d^i$
(iii) optimise $f$ along $d^i$ to find optimal “$a$” in
\[ x^{i+1} = x^i + a \cdot d^i \]
(iv) set $i = i + 1$ and iterate

Major challenges
- how to handle the large number of variables involved
- how to handle active/inactive constraints
- how to develop “feasible” directions of improvement
Graphical presentation

$x_0 \rightarrow x^1$: optimisation along $d^0$

$x_1 \rightarrow x_{r}^1$: restoration to feasible region by solving

$$h(x^1) \rightarrow h(x_{r}^1) = 0$$

d$^0$, d$^1$, ... d$^n$: optimising directions

x$^0$, x$^1$, x$_{r}^1$, x$^2$, ...: iteration points
Reduced gradient method

Basic idea:
Use the set of active constraints to eliminate the variables of the optimisation problem and turn it to an unconstrained problem.

Example problem:
A 150-variable problem with 145 equality constraints. Use the 145 constraints to eliminate 145 variables. This reduces the problem into a 5-variable unconstrained problem.
General case

\[
\min f(x)
\]

Subject to
\[
h(x) = 0
\]

Then
\[
df = \frac{\partial f}{\partial x_I} \cdot dx_I + \frac{\partial f}{\partial x_D} \cdot dx_D
\]

Reduced gradient

\[
\frac{df}{dx_I} = \frac{\partial f}{\partial x_I} + \frac{\partial f}{\partial x_D} \cdot \frac{dx_I}{dx_D}
\]

\[
dh = \left( \frac{\partial h}{\partial x_I} \right) \cdot dx_I + \left( \frac{\partial h}{\partial x_D} \right) \cdot dx_D = 0
\]

\[
\frac{df}{dx_I} = \frac{\partial f}{\partial x_I} - \left( \frac{\partial h}{\partial x_I} \right) \left( \frac{\partial h}{\partial x_D} \right)^{-1} \frac{\partial f}{\partial x_D}
\]

\[
x = [x_I, x_D]
\]

independent   dependent
min \( x_1^2 - 2x_2 \)

Subject to
\[ 3x_1 + 4x_2 = 24 \]

Let \( x_I = x_1, x_D = x_2 \) Then
\[
\frac{df}{dx_I} = \frac{df}{dx_1} = \frac{\partial f}{\partial x_1} - \left( \frac{\partial h}{\partial x_1} \right) \left( \frac{\partial h}{\partial x_2} \right)^{-1} \frac{\partial f}{\partial x_2} =
\]
\[
= 2x_1 - 3 \cdot (4)^{-1}(-2) =
\]
\[
= 2x_1 + \frac{3}{2} = d_1
\]

Minimise along \( d_1 \Rightarrow x_1^* = -\frac{3}{4} \)
Reduced gradient: general comments

• Very slow convergence, specially due to restoration

• Advance versions (without restoration) use the Lagrangian function

\[ L(x, \lambda) = f(x) + \sum_{L=1}^{n} \lambda_i h_i(x) + \sum_{i=1}^{n} p_i h_i^2(x) \]

where p_i's are penalties for violation

• Commercial software (GRG2, GINO, MINOS) includes several established packages
High performance optimisation

Penalty methods

Motivation: Make the constrained problem the unconstrained by assigning high cost on the violation of constraints

\[
\begin{align*}
\min f(x) & \quad \text{subject to } h(x) = 0 \\
\min f(x) + p \cdot h(x) & \quad \text{where } p \text{ is very large}
\end{align*}
\]

\[
\begin{align*}
\min f(x) & \quad \text{subject to } g(x) \leq 0 \\
\min f(x) + p \cdot c(x) & \quad \text{where } c(x) = \max[0, g(x)]^2
\end{align*}
\]
Barrier methods

Motivation: Make problem unconstrained by adding terms to the objective which results in high costs if the boundary is approached

\[
\begin{align*}
\min f(x) \quad \text{st} \quad g(x) \leq 0 \\
\Rightarrow \quad \min f(x) - \frac{1}{p} \frac{1}{g(x)}
\end{align*}
\]

\(p: \text{very large}\)

Advantages: They can yield superb acceleration in very large-scale formulations (i.e. defense application, weather, airline scheduling) where conventional methods suffer (*Karmakar’s method, XPRESS*)

Disadvantages: Not robust and always uncertain to converge with success