Follow the gradient

An introduction to mathematical optimisation

Gianluca Campanella

27th April 2018

Contents

Optimisation

What is optimisation?

Have An objective function, e.g. $f : \mathbb{R}^p \to \mathbb{R}$

Want The optimal $\mathbf{x}^\star$ that minimises (or maximises) f

Why?

- f represents some goal, e.g. error to be minimised
- Want the 'best' element from some set of available alternatives

Optimisation in ML

- Many ML methods are defined in terms of a loss function

$\rightarrow$ Really optimisation problems!

Optimisation in ML

- Many ML methods are defined in terms of a loss function

→ Really optimisation problems!

Linear regression

$$\mathrm{MSE}\left(\hat{\boldsymbol{\beta}} \middle| \mathbf{X}, \mathbf{y}\right) = \frac{1}{n} \sum_i \left(\hat{y}_i - y_i\right)^2$$

$$\hat{y}_i = \mathbf{x}_i \hat{\boldsymbol{\beta}}$$

Optimisation in ML

- Many ML methods are defined in terms of a loss function

→ Really optimisation problems!

Logistic regression

$$\text{LogLoss}\left(\hat{\boldsymbol{\beta}} \,\middle|\, \mathbf{X}, \mathbf{y}\right) = -\sum_i \left[y_i \log \hat{p}_i + (1 - y_i) \log(1 - \hat{p}_i)\right]$$

$$\hat{p}_i = \text{logit}^{-1}(\mathbf{x}_i \hat{\boldsymbol{\beta}})$$

Types of optimisation problems

$$f_1(\mathbf{x}) \in \mathbb{R}, \quad \mathbf{x} \in \mathbb{R}^{100}$$

$$f_2(\mathbf{x}) \in \mathbb{R}, \quad \mathbf{x} \in \mathbb{R}^{100}, \quad \mathbf{1}^\top \mathbf{x} = 1$$

$$f_3(\mathbf{x}) \in \mathbb{R}, \quad \mathbf{x} \in \{0, 1\}^{100}$$

Question

Which is 'harder' to optimise, and why?

Standard form

$$
\begin{aligned}
\min_{\mathbf{x}} \quad & f(\mathbf{x}) \\
\text{s.t.} \quad & g_j(\mathbf{x}) \leq 0, \quad j = 1, \dots, m \\
& h_k(\mathbf{x}) = 0, \quad k = 1, \dots, n \\
& l_i \leq x_i \leq u_i, \quad i = 1, \dots, p
\end{aligned}
$$

- x can be continuous or discrete
- f can be linear or nonlinear, explicit or implicit

Combinatorial optimisation

- Combinatorial problems like optimising f_3 are intrinsically hard

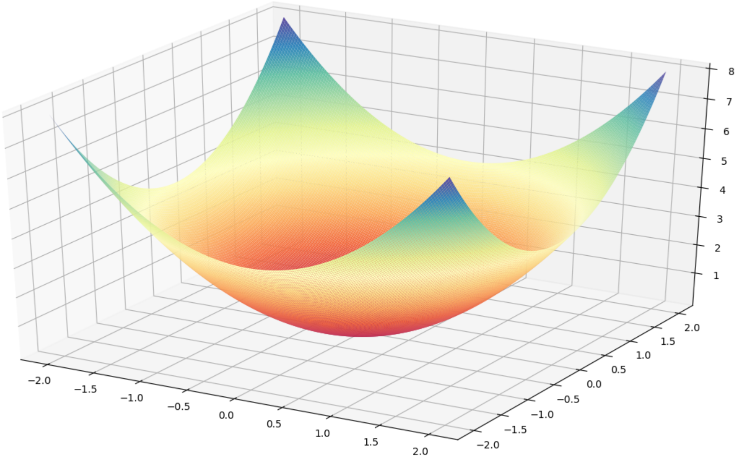
$\rightarrow$ Need to try all $2^{100} \approx 1.27 \times 10^{30}$ combinations

Side note

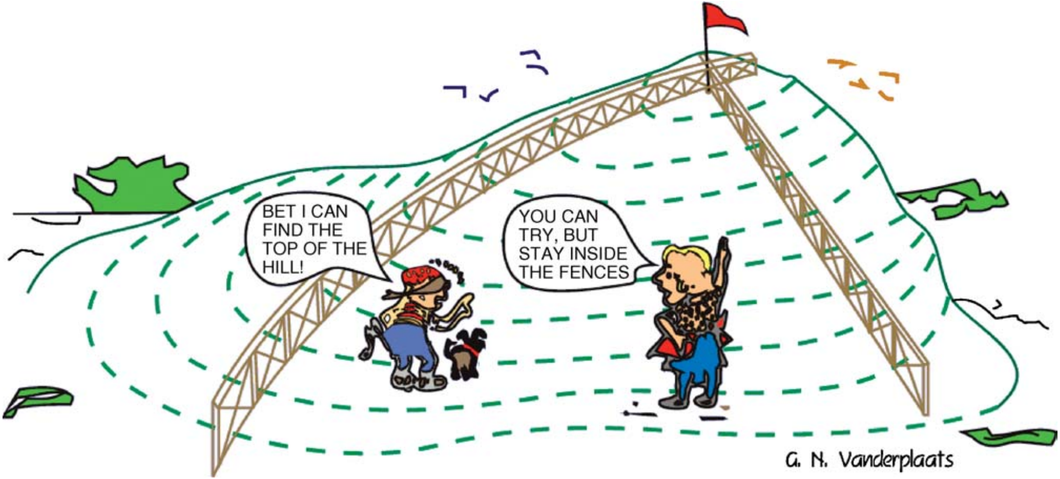
- Solving for $\mathbf{x} \in [0,1]^{100}$ is easier (assuming h is continuous)

$\rightarrow$ Approximate solution (relaxation)

Continuous optimisation

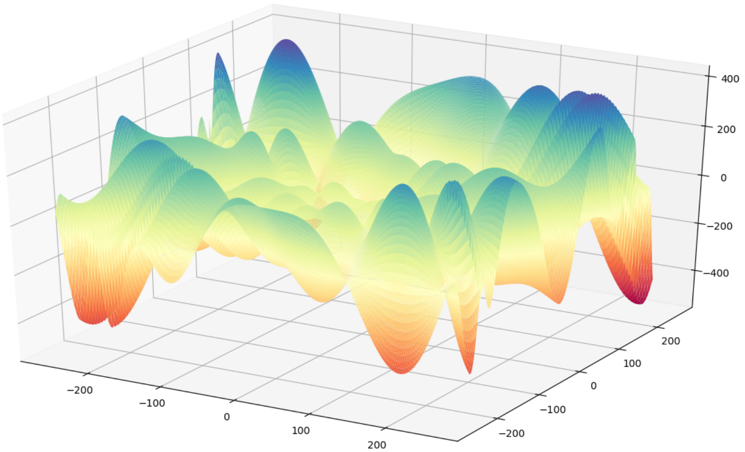


Continuous optimisation



From G. Venter (originally from G. N. Vanderplaats)

Continuous optimisation



Convex functions

Function is convex

↓

Any local minimum is also a global minimum

Linear programming

Linear programs

$$
\begin{aligned}
\max_{\mathbf{x}} \quad & \mathbf{c}^\top \mathbf{x} \\
s.t. \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\
& \mathbf{x} \geq \mathbf{0}
\end{aligned}
$$

Properties

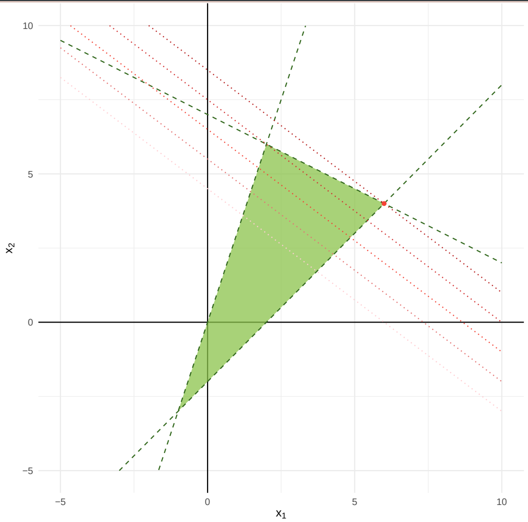
- Linear objective
- Linear constraints

Types of solution

- Optimal
- Infeasible
- Unbounded

Graphical solution

$$
\begin{aligned}
\max_{\mathbf{x}} \quad & 3x_1 + 4x_2 \\
s.t. \quad & x_1 + 2x_2 \leq 14 \\
& 3x_1 - x_2 \geq 0 \\
& x_1 - x_2 \leq 2
\end{aligned}
$$



LAD regression problem

We can rewrite the LAD (robust) regression problem

$$\min_{\boldsymbol{\beta}} \ \|\mathbf{X}\boldsymbol{\beta} - \mathbf{y}\|_1 = \sum_i |\varepsilon_i|$$

as the linear program

$$\begin{aligned} \min_{\boldsymbol{\beta},\,\mathbf{t}} \quad & \mathbf{1}_n^\top \mathbf{t} \\ s.t. \quad & -\mathbf{t} \le \mathbf{X}\boldsymbol{\beta} - \mathbf{y} \le \mathbf{t} \\ & \mathbf{t} \in \mathbb{R}^n \end{aligned}$$

or

$$\begin{aligned} \min_{\boldsymbol{\beta},\,\mathbf{u},\,\mathbf{v}} \quad & \mathbf{1}_n^\top \mathbf{u} + \mathbf{1}_n^\top \mathbf{v} \\ s.t. \quad & \mathbf{X}\boldsymbol{\beta} + \mathbf{u} - \mathbf{v} = \mathbf{y} \\ & \mathbf{u}, \mathbf{v} \ge \mathbf{0} \end{aligned}$$

τ^{th} quantile regression problem

$$\begin{aligned} \min_{\boldsymbol{\beta},\, \mathbf{u},\, \mathbf{v}} \quad & \tau\, \mathbf{1}_n^\top \mathbf{u} + (1-\tau)\mathbf{1}_n^\top \mathbf{v}, \quad \tau \in [0,1] \\ s.t. \quad & \mathbf{X}\boldsymbol{\beta} + \mathbf{u} - \mathbf{v} = \mathbf{y} \\ & \mathbf{u}, \mathbf{v} \geq \mathbf{0} \end{aligned}$$

- $\tau = 0.5$ recovers the LAD regression problem
- Very efficient (custom) algorithms exist

Convex programming

Convex quadratic programs

$$\begin{aligned} \min_{\mathbf{x}} \quad & \frac{1}{2}\mathbf{x}^\top \mathbf{Q}\mathbf{x} + \mathbf{c}^\top \mathbf{x} \\ s.t. \quad & \mathbf{A}\mathbf{x} \preceq \mathbf{b} \\ & \mathbf{x} \succeq \mathbf{0} \end{aligned}$$

Properties

- Quadratic objective
- Quadratic constraints

Question

Does quadratic imply convex?

OLS regression problem

We can rewrite the OLS regression problem

$$\min_{\boldsymbol{\beta}} \|\mathbf{X}\boldsymbol{\beta} - \mathbf{y}\|_2^2 = \sum_i \varepsilon_i^2$$

as the convex quadratic objective

$$f(\boldsymbol{\beta}) = \boldsymbol{\beta}^\top \mathbf{X}^\top \mathbf{X}\boldsymbol{\beta} - 2\mathbf{y}^\top \mathbf{X}\boldsymbol{\beta} + \mathbf{y}^\top \mathbf{y}$$

OLS regression problem

Setting the gradient to 0 and solving for β...

$$\begin{aligned}
\nabla f &= 2\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta} - 2\mathbf{X}^\top\mathbf{y} = 0 \\
\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta} &= \mathbf{X}^\top\mathbf{y} \\
\hat{\boldsymbol{\beta}} &= \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{y}
\end{aligned}$$

Ridge regularisation

$$\min_{\boldsymbol{\beta}} \|\mathbf{X}\boldsymbol{\beta} - \mathbf{y}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2, \quad \lambda \geq 0$$

The objective becomes…

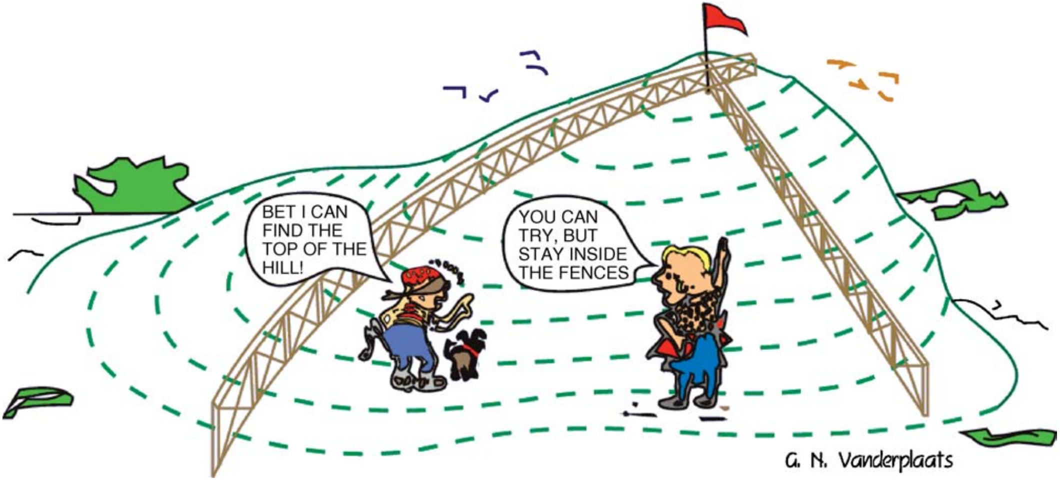
$$f(\boldsymbol{\beta}) = \boldsymbol{\beta}^\top \left(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I}_p\right)\boldsymbol{\beta} - 2\mathbf{y}^\top\mathbf{X}\boldsymbol{\beta} + \mathbf{y}^\top\mathbf{y}$$

Constraints on β

Condition	Useful for…
$\boldsymbol{\beta} \geq \mathbf{0}$	Intensities or rates
$\mathbf{l} \leq \boldsymbol{\beta} \leq \mathbf{u}$	Knowledge of permissible values
$\boldsymbol{\beta} \geq \mathbf{0} \wedge \mathbf{1}_p^\top \boldsymbol{\beta} = 1$	Proportions and probability distributions

Follow the gradient

Why follow the gradient?



From G. Venter (originally from G. N. Vanderplaats)

Karush-Kuhn-Tucker conditions

1. $\mathbf{x}^\star$ is feasible

2. The gradient of the Lagrangian vanishes at $\mathbf{x}^\star$

$$\nabla f(\mathbf{x}^\star) + \sum_{j=1}^{m} \lambda_j \nabla g_j(\mathbf{x}^\star) + \sum_{k=1}^{n} \lambda_{m+k} \nabla h_k(\mathbf{x}^\star) = \mathbf{0}, \quad \lambda_j \geq 0, \quad \lambda_{m+k} \in \mathbb{R}$$

3. For each inequality constraint,

$$\lambda_j g_j(\mathbf{x}^\star) = 0, \quad j = 1, \dots, m$$

General idea

$$\mathbf{x} \mapsto \mathbf{x} + \alpha^\star \, \mathbf{s}$$

1. Find a search direction $\mathbf{s}$ in which to move
2. Take the optimal step size $\alpha^\star$ in this direction

General idea

$$\mathbf{x} \mapsto \mathbf{x} + \alpha^\star \mathbf{s}$$

1. Find a search direction $\mathbf{s}$ in which to move
2. Take the optimal step size $\alpha^\star$ in this direction

Gradient calculation

- Pen and paper
- Finite differences
- Automatic differentiation

Finite differences

Good

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

- One function call
- Error: $\mathcal{O}(h)$

Better

$$f'(x) \approx \frac{f(x+h/2) - f(x-h/2)}{h}$$

- Two function calls
- Error: $\mathcal{O}(h^2)$

Automatic differentiation

The derivative of the composition

$$f \circ g \circ h(x) = f(g(h(x)))$$

is given by the chain rule

$$\frac{d(f \circ g \circ h)}{dx} = \frac{df}{dg}\frac{dg}{dh}\frac{dh}{dx} = \left[\frac{df}{dg}\left(\frac{dg}{dh}\frac{dh}{dx}\right)\right] = \left[\left(\frac{df}{dg}\frac{dg}{dh}\right)\frac{dh}{dx}\right]$$

Automatic differentiation

The derivative of the composition

$$f \circ g \circ h(x) = f(g(h(x)))$$

is given by the chain rule

$$\frac{d(f \circ g \circ h)}{dx} = \frac{df}{dg}\frac{dg}{dh}\frac{dh}{dx} = \left[\frac{df}{dg}\left(\frac{dg}{dh}\frac{dh}{dx}\right)\right] = \left[\left(\frac{df}{dg}\frac{dg}{dh}\right)\frac{dh}{dx}\right]$$

Forward-mode differentiation

$$f(x,y) = 3x^2 + xy \qquad \frac{\partial f}{\partial x} = 6x + y \qquad \frac{\partial f}{\partial y} = x$$

$$\begin{aligned}
x &= ? & \partial x/\partial \square &= ? \\
y &= ? & \partial y/\partial \square &= ? \\
a &= x^2 & \partial a/\partial \square &= 2x \times \partial x/\partial \square \\
b &= 3 \times a & \partial b/\partial \square &= 3 \times \partial a/\partial \square \\
c &= x \times y & \partial c/\partial \square &= y \times \partial x/\partial \square + x \times \partial y/\partial \square \\
f &= b + c & \partial f/\partial \square &= \partial b/\partial \square + \partial c/\partial \square
\end{aligned}$$

Forward-mode differentiation

$$f(x,y) = 3x^2 + xy \qquad \frac{\partial f}{\partial x} = 6x + y \qquad \frac{\partial f}{\partial y} = x$$

$$\partial x / \partial x = 1$$
$$\partial y / \partial x = 0$$
$$\partial a / \partial x = 2x \times \partial x / \partial x = 2x$$
$$\partial b / \partial x = 3 \times \partial a / \partial x = 6x$$
$$\partial c / \partial x = y \times \partial x / \partial x + x \times \partial y / \partial x = y$$
$$\partial f / \partial x = \partial b / \partial x + \partial c / \partial x = 6x + y$$

$$\partial x / \partial y = 0$$
$$\partial y / \partial y = 1$$
$$\partial a / \partial y = 2x \times \partial x / \partial y = 0$$
$$\partial b / \partial y = 3 \times \partial a / \partial y = 0$$
$$\partial c / \partial y = y \times \partial x / \partial y + x \times \partial y / \partial y = x$$
$$\partial f / \partial y = \partial b / \partial y + \partial c / \partial y = x$$

Reverse-mode differentiation

$$f(x, y) = 3x^2 + xy \qquad \frac{\partial f}{\partial x} = 6x + y \qquad \frac{\partial f}{\partial y} = x$$

$$\begin{aligned}
\partial x / \partial \Box &= ? \\
\partial y / \partial \Box &= ? \\
\partial a / \partial \Box &= 2x \times \partial x / \partial \Box \\
\partial b / \partial \Box &= 3 \times \partial a / \partial \Box \\
\partial c / \partial \Box &= y \times \partial x / \partial \Box + x \times \partial y / \partial \Box \\
\partial f / \partial \Box &= \partial b / \partial \Box + \partial c / \partial \Box
\end{aligned}$$

$$\begin{aligned}
\partial \Diamond / \partial f &= ? \\
\partial \Diamond / \partial c &= \partial \Diamond / \partial f \\
\partial \Diamond / \partial b &= \partial \Diamond / \partial f \\
\partial \Diamond / \partial a &= 3 \times \partial \Diamond / \partial b \\
\partial \Diamond / \partial y &= x \times \partial \Diamond / \partial f \\
\partial \Diamond / \partial x &= 2x \times \partial \Diamond / \partial a + y \times \partial \Diamond / \partial c
\end{aligned}$$

Reverse-mode differentiation

$$f(x, y) = 3x^2 + xy \qquad \frac{\partial f}{\partial x} = 6x + y \qquad \frac{\partial f}{\partial y} = x$$

$$\partial f / \partial f = 1$$

$$\partial f / \partial c = \partial f / \partial f = 1$$

$$\partial f / \partial b = \partial f / \partial f = 1$$

$$\partial f / \partial a = 3 \times \partial f / \partial b = 3$$

$$\partial f / \partial y = x \times \partial f / \partial f = x$$

$$\partial f / \partial x = 2x \times \partial f / \partial a + y \times \partial f / \partial c = 6x + y$$

Newton’s method

f can be approximated about an initial guess $\mathbf{x}_0$ as

$$f(\mathbf{x}) \approx f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^\top (\mathbf{x} - \mathbf{x}_0) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_0)^\top H(\mathbf{x}_0)(\mathbf{x} - \mathbf{x}_0)$$

Newton's method

We want to find $\delta = \mathbf{x}^\star - \mathbf{x}_0$ such that $\nabla f(\mathbf{x}^\star) = \mathbf{0}$

$$\nabla_\delta \tilde{f} = \nabla f(\mathbf{x}_0) + H(\mathbf{x}_0)\,\delta = \mathbf{0}$$
$$\delta = -H^{-1}(\mathbf{x}_0)\,\nabla f(\mathbf{x}_0)$$

This gives the update

$$\mathbf{x} \mapsto \mathbf{x} + \delta = \mathbf{x} - H^{-1}(\mathbf{x})\,\nabla f(\mathbf{x})$$

Quasi-Newton methods

- $H^{-1}(\mathbf{x})$ may be large and expensive to compute

→ Use an approximation

Gradient descent

Forget about it

$$H^{-1}(\mathbf{x}) \approx \mathbf{I}_p$$

BFGS and L-BFGS

Update iteratively

$$B_i \delta = -\nabla f(\mathbf{x}_i)$$

Stochastic gradient descent

Many ML methods are sum-minimisation problems

$$\min_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) = \sum_i f_i(\boldsymbol{\theta})$$

This means the update $\boldsymbol{\theta} \mapsto \boldsymbol{\theta} - \alpha^\star \nabla f(\boldsymbol{\theta})$ is actually

$$\boldsymbol{\theta} \mapsto \boldsymbol{\theta} - \alpha^\star \sum_i \nabla f_i(\boldsymbol{\theta})$$

Stochastic gradient descent

1. Shuffle observations
2. $\boldsymbol{\theta} \mapsto \boldsymbol{\theta} - \alpha^\star \nabla f_i(\boldsymbol{\theta})$ for each observation i → one pass
3. Repeat until convergence

How do we choose $\alpha^\star$?

Large α $\rightarrow$ Divergence

Small α $\rightarrow$ Slow convergence

- Decrease α in later iterations
- Use a linear combination with the previous update (momentum)
- Average θ over iterations
- Use per-parameter step sizes