Mathematical Concepts 1

Exercise 1: Gradient

Consider the bivariate function $f : \mathbb{R}^2 \to \mathbb{R}, (x_1, x_2) \mapsto x_1^2 + 0.5x_2^2 + x_1x_2$.

(a) Show that f is smooth (as defined in the lecture).

(b) Find the direction of greatest increase of f at $\mathbf{x} = (1, 1)$.

(c) Find the direction of greatest decrease of f at $\mathbf{x} = (1, 1)$.

(d) Find a direction in which f does not instantly change at $\mathbf{x} = (1, 1)$.

(e) Assume there exists a differentiable parametrization of a curve $\tilde{\mathbf{x}} : \mathbb{R} \to \mathbb{R}^2, t \mapsto \tilde{\mathbf{x}}(t)$ such that $\forall t \in \mathbb{R} : f(\tilde{\mathbf{x}}(t)) = f(1, 1)$. Show that at each point of the curve $\tilde{\mathbf{x}}$ the tangent line $\frac{\partial \tilde{\mathbf{x}}}{\partial t}$ is perpendicular to the gradient $\nabla f(\tilde{\mathbf{x}})$.

(f) Interpret (d), (e) geometrically

Exercise 2: Convexity

Consider two convex functions $f, g : \mathbb{R} \to \mathbb{R}$.

(a) Show that $f + g : \mathbb{R} \to \mathbb{R}, x \mapsto f(x) + g(x)$ is convex.

(b) Now, assume that g is additionally non-decreasing, i.e., $g(y) \geq g(x)\ \forall x \in \mathbb{R}, \forall y \in \mathbb{R}$ with $y > x$. Show that $g \circ f$ is convex.

Exercise 3: Taylor polynomials

Consider the bivariate function $f : \mathbb{R}^2 \to \mathbb{R}, (x_1, x_2) \mapsto \exp(\pi \cdot x_1) - \sin(\pi \cdot x_2) + \pi \cdot x_1 \cdot x_2$

(a) Compute the gradient of f for an arbitrary $\mathbf{x}$.

(b) Compute the Hessian of f for an arbitrary $\mathbf{x}$.

(c) State the first order taylor polynomial $T_{1,\mathbf{a}}(\mathbf{x})$ expanded around the point $\mathbf{a} = (0, 1)$.

(d) State the second order taylor polynomial $T_{2,\mathbf{a}}(\mathbf{x})$ expanded around the point $\mathbf{a} = (0, 1)$.

(e) Determine if $T_{2,\mathbf{a}}$ is a convex function.

Mathematical Concepts 1

Solution 1:
Gradient

(a) The gradient $\nabla f(\mathbf{x}) = (2x_1 + x_2, x_2 + x_1)^\top$ is continuous $\Rightarrow f \in \mathcal{C}^1$.

(b) The direction of greatest increase is given by the gradient, i.e., $\nabla f(1,1) = (3,2)^\top$.

(c) Let $\mathbf{v} \in \mathbb{R}^2$ be a direction with fixed length $\|\mathbf{v}\|_2 = r > 0$.
The directional derivative $D_\mathbf{v} f(\mathbf{x}) = \nabla f(\mathbf{x})^\top \mathbf{v} = \|\nabla f(\mathbf{x})\|_2 \|\mathbf{v}\|_2 \cos(\theta) = \|\nabla f(\mathbf{x})\|_2 r \cos(\theta)$. This becomes minimal if $\theta = \pi$, i.e., if $\mathbf{v}$ points in the opposite direction of $\nabla f \Rightarrow \mathbf{v} = -\nabla f(\mathbf{x})$ if $r = \|\nabla f(\mathbf{x})\|_2$. Here, the direction of greatest decrease is given by $-\nabla f(1,1) = (-3,-2)^\top$.

(d) $D_\mathbf{v} f(\mathbf{x}) = \nabla f(1,1)^\top \mathbf{v} \overset{!}{=} 0 \Rightarrow (3,2) \cdot \mathbf{v} = 0 \iff \mathbf{v} = \alpha \cdot (-2,3)^\top$ with $\alpha \in \mathbb{R}$ and $\alpha \neq 0$.

(e) When we differentiate both sides of the equation $f(\tilde{\mathbf{x}}(t)) = f(1,1)$ w.r.t. t we arrive at $\frac{\partial f(\tilde{\mathbf{x}}(t))}{\partial t} = 0$. Via the chain rule it follows that $\underbrace{\frac{\partial f}{\partial \tilde{\mathbf{x}}}}_{= \nabla f(\tilde{\mathbf{x}})^\top} \frac{\partial \tilde{\mathbf{x}}}{\partial t} = 0$.

(f) The gradient is orthogonal to the tangent line of the level curves.

Solution 2:
Convexity

(a) Let $x, y \in \mathbb{R}$ and $t \in [0,1]$ then it holds that

$$\begin{aligned}
(f+g)(x + t(y-x)) &= f(x + t(y-x)) + g(x + t(y-x)) \\
&\leq f(x) + t(f(y) - f(x)) + g(x) + t(g(y) - g(x)) && (f, g \text{ are convex}) \\
&= f(x) + g(x) + t(f(y) + g(y) - (f(x) + g(x))) \\
&= (f+g)(x) + t((f+g)(y) - (f+g)(x)).
\end{aligned}$$

(b) Let $x, y \in \mathbb{R}$ and $t \in [0,1]$ then it holds that

$$\begin{aligned}
(g \circ f)(x + t(y-x)) &= g(f(x + t(y-x))) \\
&\leq g(f(x) + t(f(y) - f(x))) && (g \text{ is non-decreasing, } f \text{ is convex}) \\
&\leq g(f(x)) + t(g(f(y)) - g(f(x)))) && (g \text{ is convex}) \\
&= (g \circ f)(x) + t((g \circ f)(y) - (g \circ f)(x)).
\end{aligned}$$

Solution 3:
Convexity

Consider the bivariate function $f : \mathbb{R}^2 \to \mathbb{R}, (x_1, x_2) \mapsto \exp(\pi \cdot x_1) - \sin(\pi \cdot x_2) + \pi \cdot x_1 \cdot x_2$

(a) $\nabla f(\mathbf{x}) = \pi \cdot (\exp(\pi x_1) + x_2, -\cos(\pi x_2) + x_1)^\top$

(b) $\nabla^2 f(\mathbf{x}) = \pi \cdot \begin{pmatrix} \pi \exp(\pi x_1) & 1 \\ 1 & \pi \sin(\pi x_2) \end{pmatrix}$

(c) $T_{1,\mathbf{a}}(\mathbf{x}) = f(\mathbf{a}) + \nabla f(\mathbf{a})^\top (\mathbf{x} - \mathbf{a}) = 1 + \pi \cdot (2,1) \cdot (x_1, x_2 - 1)^\top = 1 - \pi + 2\pi x_1 + \pi x_2$

(d)

$$\begin{aligned} T_{2,\mathbf{a}}(\mathbf{x}) &= T_{1,\mathbf{a}}(\mathbf{x}) + \frac{1}{2}(\mathbf{x}-\mathbf{a})^\top \nabla^2 f(\mathbf{a})(\mathbf{x}-\mathbf{a}) \\ &= T_{1,\mathbf{a}}(\mathbf{x}) + \frac{1}{2}\mathbf{x}^\top \nabla^2 f(\mathbf{a})\mathbf{x} + \mathbf{x}^\top \nabla^2 f(\mathbf{a})\mathbf{a} + \frac{1}{2}\mathbf{a}^\top \nabla^2 f(\mathbf{a})\mathbf{a} \end{aligned}$$

With $\nabla^2 f(\mathbf{a}) = \begin{pmatrix} \pi^2 & \pi \\ \pi & 0 \end{pmatrix}$ we get that

$$\begin{aligned} T_{2,\mathbf{a}}(\mathbf{x}) = T_{1,\mathbf{a}}(\mathbf{x}) &+ 0.5\pi^2 x_1^2 \\ &+ \pi x_1 x_2 - \pi x_1 \\ &+ 0. \end{aligned}$$

.

(e) $T_{2,\mathbf{a}}(\mathbf{x})$ is multivariate polynomial of degree 2 which means its Hessian is constant and we can directly see that $\mathbf{H} := \nabla^2 T_{2,\mathbf{a}}(\mathbf{x}) = \nabla^2 f(\mathbf{a})$. For the eigenvalues of the Hessian it must hold that

$$\begin{aligned} & \det(\mathbf{H} - \lambda \mathbf{I}) && = 0 \\ \iff & \det \begin{pmatrix} \pi^2 - \lambda & \pi \\ \pi & -\lambda \end{pmatrix} && = 0 \\ \iff & (\pi^2 - \lambda)\cdot(-\lambda) - \pi^2 && = 0 \\ \iff & \lambda^2 - \pi^2\lambda - \pi^2 && = 0. \end{aligned}$$

From which it follows that $\lambda_{1,2} = \frac{\pi^2 \pm \sqrt{\pi^4 + 4\pi^2}}{2} \Rightarrow \lambda_1 \approx 10.785, \lambda_2 \approx -0.915$. Since $\lambda_2 < 0$ $T_{2,\mathbf{a}}$ is not convex.

Mathematical Concepts 2

Exercise 1: Matrix Calculus

Let $\mathbf{x}, \mathbf{c} \in \mathbb{R}^d, \mathbf{u}, \mathbf{v} : \mathbb{R}^d \to \mathbb{R}^d, \mathbf{u}, \mathbf{v} \in \mathcal{C}^2, \mathbf{Y} : \mathbb{R}^d \to \mathbb{R}^{d \times d}$

(a) Compute $\frac{\partial \|\mathbf{x}-\mathbf{c}\|_2^2}{\partial \mathbf{x}}$

(b) Compute $\frac{\partial \|\mathbf{x}-\mathbf{c}\|_2}{\partial \mathbf{x}}$

(c) Compute $\frac{\partial \mathbf{u}^\top \mathbf{v}}{\partial \mathbf{x}}$

(d) Show that $\frac{\partial \mathbf{Y}^\top \mathbf{u}}{\partial \mathbf{x}} = \begin{pmatrix} \mathbf{y}_1^\top \frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \mathbf{u}^\top \frac{\partial \mathbf{y}_1}{\partial \mathbf{x}} \\ \vdots \\ \mathbf{y}_d^\top \frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \mathbf{u}^\top \frac{\partial \mathbf{y}_d}{\partial \mathbf{x}} \end{pmatrix}$ where $\mathbf{y}_i : \mathbb{R}^d \to \mathbb{R}^d, i = 1, \ldots, d$ are the column vectors of $\mathbf{Y}$.

(e) Compute $\frac{\partial^2 \mathbf{u}^\top \mathbf{v}}{\partial \mathbf{x} \partial \mathbf{x}^\top}$
Hint: use c), d)

Exercise 2: Optimality in 1d

Let $f : [-1, 2] \to \mathbb{R}, x \mapsto \exp(x^3 - 2x^2)$

(a) Compute f'

(b) Plot f and f' with `R`

(c) Find all possible candidates x^* for maxima and minima.
Hint: exp is a strictly monotone function.

(d) Compute f''

(e) Determine if the candidates are local maxima, minima or neither.

(f) Find the global maximum and global minimum of f

Mathematical Concepts 2

Solution 1:
Matrix Calculus

(a) $\frac{\partial \|\mathbf{x}-\mathbf{c}\|_2^2}{\partial \mathbf{x}} = \frac{\partial \|\mathbf{u}\|_2^2}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial \mathbf{x}} = \frac{\partial \mathbf{u}^\top \mathbf{u}}{\partial \mathbf{u}} \frac{\partial \mathbf{x}-\mathbf{c}}{\partial \mathbf{x}} = \frac{\partial \mathbf{u}^\top \mathbf{I}\mathbf{u}}{\partial \mathbf{u}}(\mathbf{I}-\mathbf{0}) = \mathbf{u}^\top(\mathbf{I}+\mathbf{I}^\top) = 2(\mathbf{x}-\mathbf{c})^\top$

(b) $\frac{\partial \|\mathbf{x}-\mathbf{c}\|_2}{\partial \mathbf{x}} = \frac{\partial \sqrt{\|\mathbf{x}-\mathbf{c}\|_2^2}}{\partial \mathbf{x}} = \frac{0.5}{\sqrt{\|\mathbf{x}-\mathbf{c}\|_2^2}} \frac{\partial \|\mathbf{x}-\mathbf{c}\|_2^2}{\partial \mathbf{x}} \overset{(a)}{=} \frac{(\mathbf{x}-\mathbf{c})^\top}{\|\mathbf{x}-\mathbf{c}\|_2}$

(c) $\frac{\partial \mathbf{u}^\top \mathbf{v}}{\partial \mathbf{x}} = \frac{\partial \mathbf{u}^\top \mathbf{I}\mathbf{v}}{\partial \mathbf{x}} = \mathbf{u}^\top \mathbf{I} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} + \mathbf{v}^\top \mathbf{I}^\top \frac{\partial \mathbf{u}}{\partial \mathbf{x}} = \mathbf{u}^\top \frac{\partial \mathbf{v}}{\partial \mathbf{x}} + \mathbf{v}^\top \frac{\partial \mathbf{u}}{\partial \mathbf{x}}$

(d) $\frac{\partial \mathbf{Y}^\top \mathbf{u}}{\partial \mathbf{x}} = \frac{\partial \begin{pmatrix} \mathbf{y}_1^\top \mathbf{u} \\ \vdots \\ \mathbf{y}_d^\top \mathbf{u} \end{pmatrix}}{\partial \mathbf{x}} \overset{(c)}{=} \begin{pmatrix} \mathbf{y}_1^\top \frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \mathbf{u}^\top \frac{\partial \mathbf{y}_1}{\partial \mathbf{x}} \\ \vdots \\ \mathbf{y}_d^\top \frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \mathbf{u}^\top \frac{\partial \mathbf{y}_d}{\partial \mathbf{x}} \end{pmatrix}$

(e) Note for $\mathbf{y} : \mathbb{R}^d \to \mathbb{R}^d, \mathbf{x} \mapsto \mathbf{y}(\mathbf{x})$ the $i-$th column of $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ is $\frac{\partial \mathbf{y}}{\partial x_i}$. With this it follows that

$$
\begin{aligned}
\frac{\partial^2 \mathbf{u}^\top \mathbf{v}}{\partial \mathbf{x} \partial \mathbf{x}^\top} &= \frac{\partial}{\partial \mathbf{x}} \left(\frac{\partial \mathbf{u}^\top \mathbf{v}}{\partial \mathbf{x}^\top} \right) \\
&= \frac{\partial}{\partial \mathbf{x}} \left[\left(\frac{\partial \mathbf{u}^\top \mathbf{v}}{\partial \mathbf{x}} \right)^\top \right] \\
&\overset{(c)}{=} \frac{\partial (\mathbf{u}^\top \frac{\partial \mathbf{v}}{\partial \mathbf{x}} + \mathbf{v}^\top \frac{\partial \mathbf{u}}{\partial \mathbf{x}})^\top}{\partial \mathbf{x}} \\
&= \frac{\partial \left(\left(\frac{\partial \mathbf{v}}{\partial \mathbf{x}}\right)^\top \mathbf{u} + \left(\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\right)^\top \mathbf{v} \right)}{\partial \mathbf{x}} \\
&\overset{(d)}{=} \begin{pmatrix} \mathbf{u}^\top \frac{\partial^2 \mathbf{v}}{\partial x_1 \partial \mathbf{x}} + \frac{\partial \mathbf{v}}{\partial x_1}^\top \frac{\partial \mathbf{u}}{\partial \mathbf{x}} \\ \vdots \\ \mathbf{u}^\top \frac{\partial^2 \mathbf{v}}{\partial x_d \partial \mathbf{x}} + \frac{\partial \mathbf{v}}{\partial x_d}^\top \frac{\partial \mathbf{u}}{\partial \mathbf{x}} \end{pmatrix}^\top + \begin{pmatrix} \mathbf{v}^\top \frac{\partial^2 \mathbf{u}}{\partial x_1 \partial \mathbf{x}} + \frac{\partial \mathbf{u}}{\partial x_1}^\top \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \\ \vdots \\ \mathbf{v}^\top \frac{\partial^2 \mathbf{u}}{\partial x_d \partial \mathbf{x}} + \frac{\partial \mathbf{u}}{\partial x_d}^\top \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \end{pmatrix}^\top \\
&= \begin{pmatrix} \mathbf{u}^\top \frac{\partial^2 \mathbf{v}}{\partial x_1 \partial \mathbf{x}} \\ \vdots \\ \mathbf{u}^\top \frac{\partial^2 \mathbf{v}}{\partial x_d \partial \mathbf{x}} \end{pmatrix}^\top + \frac{\partial \mathbf{u}}{\partial \mathbf{x}} \left(\frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^\top + \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \left(\frac{\partial \mathbf{u}}{\partial \mathbf{x}} \right)^\top + \begin{pmatrix} \mathbf{v}^\top \frac{\partial^2 \mathbf{u}}{\partial x_1 \partial \mathbf{x}} \\ \vdots \\ \mathbf{v}^\top \frac{\partial^2 \mathbf{u}}{\partial x_d \partial \mathbf{x}} \end{pmatrix}^\top
\end{aligned}
$$

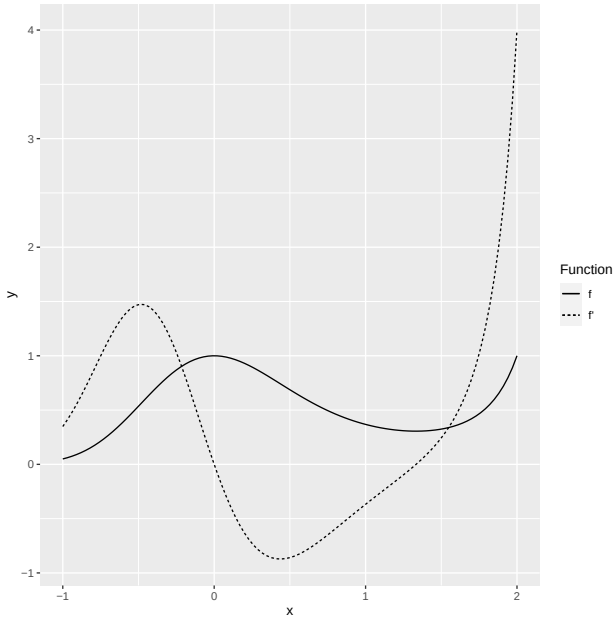
Solution 2:
Optimality in 1d

Let $f : [-1, 2] \to \mathbb{R}, x \mapsto \exp(x^3 - 2x^2)$

(a) $f'(x) = \exp(x^3 - 2x^2) \cdot (3x^2 - 4x)$

(b)
```
library(ggplot2)

f <- function(x) exp(x^3 - 2*x^2)
df <- function(x) f(x) * (3*x^2 - 4*x)

ggplot(data.frame(x = seq(-1, 2, by=0.005)), aes(x)) +
```

```
  geom_function(fun = f, aes(linetype = "f")) +
  geom_function(fun = df, aes(linetype = "f'"))    +
  scale_linetype_discrete(name = "Function")
```



(c) f is continuously differentiable $\Rightarrow$ candidates can only be stationary points and boundary points.
Find stationary points, i.e., points where
$f'(x) = 0 \iff \underbrace{\exp(x^3 - 2x^2)}_{>0} \cdot (3x^2 - 4x) = 0 \iff 3x^2 - 4x = 0 \iff x(3x - 4) = 0.$
$\Rightarrow x_1 = 0, x_2 = 4/3$. The other candidates are boundary points, i.e., $x_3 = -1, x_4 = 2$.

(d) $f''(x) = \exp(x^3 - 2x^2) \cdot (3x^2 - 4x)^2 + \exp(x^3 - 2x^2) \cdot (6x - 4)$

(e) $f''(x_1) = \exp(0) \cdot (-4) < 0$
$\Rightarrow x_1$ is a local maximum

$f''(x_2) = \exp((4/3)^3 - 2(4/3)^2) \cdot (4) > 0$
$\Rightarrow x_2$ is a local minimum.

The boundary points x_3 and x_4 are not considered as *local* optima.

(f) $f(x_1) = \exp(0) = 1$
$f(x_2) = \exp((4/3)^3 - 2(4/3)^2) \approx 0.3057$
$f(x_3) = \exp(-3) \approx 0.05$
$f(x_4) = \exp(0) = 1$
$\Rightarrow x_1, x_4$ are global maxima. x_3 is global minimum.

Mathematical Concepts 3

Exercise 1: Optimality in 2 dimensions

Let $f : \mathbb{R}^2 \to \mathbb{R}, (x_1, x_2) \mapsto -\cos(x_1^2 + x_2^2 + x_1 x_2)$

(a) Create a contour plot of f in the range $[-2, 2] \times [-2, 2]$ with `R`.

(b) Compute ∇f

(c) Compute $\nabla^2 f$

Now, we define the restriction of f to $S_r = \{(x_1, x_2) \in \mathbb{R}^2 | \ x_1^2 + x_2^2 + x_1 x_2 < r\}$ with $r \in \mathbb{R}, r > 0$, i.e., $f_{|S_r} : S_r \to \mathbb{R}, (x_1, x_2) \mapsto f(x_1, x_2)$.

(d) Show that $f_{|S_{\overline{r}}}$ with $\overline{r} = \pi/4$ is convex.

(e) Find the local minimum $\mathbf{x}^*$ of $f_{|S_{\overline{r}}}$

(f) Is $\mathbf{x}^*$ a global minimum of f?

Exercise 2: Optimality in d dimensions

Let $\mathbf{X}$ be a d-dimensional random vector and let $\mathbf{Y}$ be a one-dimensional random vector with $\mathsf{Var}(\mathbf{X}) = \mathbf{\Sigma_X}$ and $\mathsf{Cov}(\mathbf{X}, \mathbf{Y}) = \mathbf{\Sigma_{X,Y}} \in \mathbb{R}^{d \times 1}$.
Further, let $f : \mathbb{R}^d \to \mathbb{R}, \mathbf{w} \mapsto \mathsf{Var}(\mathbf{w}^\top \mathbf{X} - \mathbf{Y})$.

(a) Show that f is convex.

(b) Compute ∇f and $\nabla^2 f$

(c) Under which condition exists a unique minimizer $\mathbf{w}^*$ of f. Is this a global minimum? (if it exists)

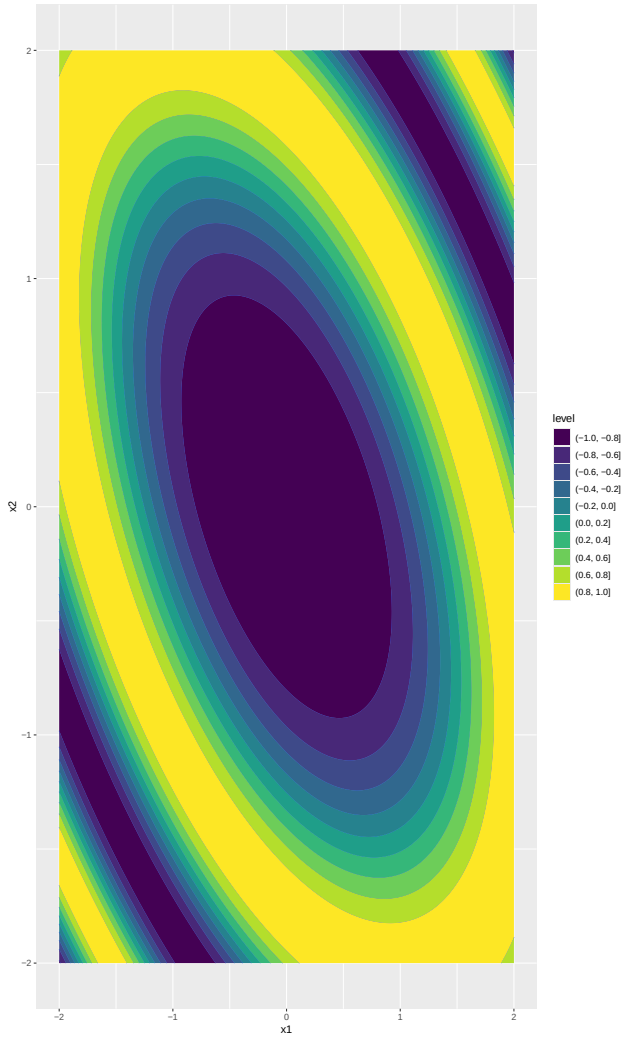
(d) Given the samples $(\mathbf{x}_i, y_i) \sim \mathbb{P}_{\mathbf{X},\mathbf{Y}}$, under which condition is the least squares estimator a consistent estimator of $\mathbf{w}^*$ in general?[1]

[1] This question is out of the scope of this lecture; however, it gives interesting insights into the entities we have computed.

Mathematical Concepts 3

Solution 1:
Optimality in 2d

(a)
```
library(ggplot2)

f <- function(x, y) - cos(x^2 + y^2 + x*y)
x = seq(-2, 2, by=0.01)
xx = expand.grid(X1 = x, X2 = x)

fxx = f(xx[, 1], xx[, 2])
df = data.frame(X1 = xx$X1, X2 = xx$X2, fxx = fxx)

ggplot(df, aes(x = X1, y = X2, z = fxx)) +
  geom_contour_filled() +
  xlab("x1") +
  ylab("x2")
```



(b) $\nabla f = (\sin(x_1^2 + x_2^2 + x_1x_2)(2x_1 + x_2), \sin(x_1^2 + x_2^2 + x_1x_2)(2x_2 + x_1))^\top$

(c) $\nabla^2 f = \begin{pmatrix} \cos(u)(2x_1 + x_2)^2 + 2\sin(u) & \cos(u)(2x_1 + x_2)(2x_2 + x_1) + \sin(u) \\ \cos(u)(2x_1 + x_2)(2x_2 + x_1) + \sin(u) & \cos(u)(2x_2 + x_1)^2 + 2\sin(u) \end{pmatrix}$ with $u = x_1^2 + x_2^2 + x_1x_2$.

(d) Let $u : \mathbb{R}^2 \to \mathbb{R}, (x_1, x_2) \mapsto x_1^2 + x_2^2 + x_1x_2$, such that $f(\mathbf{x}) = -\cos(u(\mathbf{x}))$.
$\implies \nabla^2 f(\mathbf{x}) = \cos(u(\mathbf{x}))\nabla u(\mathbf{x})\nabla u(\mathbf{x})^\top + \sin(u(\mathbf{x}))\nabla^2 u(\mathbf{x})$
$\nabla u(\mathbf{x}) = (2x_1 + x_2, x_1 + 2x_2)^\top$
$\nabla^2 u(\mathbf{x}) = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$
For $\mathbf{x} \in S_{\bar{r}}$, it holds that $u(\mathbf{x}) \geq 0$, since

$$0 \leq \frac{1}{2}(x_1 + x_2)^2 = \frac{1}{2}x_1^2 + \frac{1}{2}x_2^2 + x_1x_2 \leq x_1^2 + x_2^2 + x_1x_2 = u(\mathbf{x}),$$

and that $u(\mathbf{x}) < \pi/4$. This implies that $\cos(u(\mathbf{x})) > 0$ and $\sin(u(\mathbf{x})) \geq 0$.
$\nabla u(\mathbf{x})\nabla u(\mathbf{x})^\top$ is positive semi-definite since

$$\mathbf{v}^\top \nabla u(\mathbf{x})\nabla u(\mathbf{x})^\top \mathbf{v} = (\mathbf{v}^\top \nabla u(\mathbf{x}))^2 \geq 0.$$

$\nabla^2 u(\mathbf{x})$ is positive definite since

$$\mathbf{v}^\top \nabla^2 u(\mathbf{x})\mathbf{v} = 2v_1^2 + 2v_1v_2 + 2v_2^2 = v_1^2 + v_2^2 + (v_1 + v_2)^2 \geq 0$$

and equality only holds if $\mathbf{v} = \mathbf{0}$.
So, in total, for $\mathbf{x} \in S_{\bar{r}}$, we have that

$$\nabla^2 f(\mathbf{x}) = \underbrace{\cos(u(\mathbf{x}))}_{>0}\underbrace{\nabla u(\mathbf{x})\nabla u(\mathbf{x})^\top}_{\text{p.s.d.}} + \underbrace{\sin(u(\mathbf{x}))}_{\geq 0}\underbrace{\nabla^2 u(\mathbf{x})}_{\text{p.d.}}.$$

$\Rightarrow \nabla^2 f(\mathbf{x})$ is positive semi-definite.
$\Rightarrow f_{|S_{\bar{r}}}$ is convex.

(e) For $\mathbf{x} \in S_{\bar{r}}$, we have

$$\nabla f(\mathbf{x}) = \sin(u(\mathbf{x}))\, \nabla u(\mathbf{x}),$$

where $\sin(u(\mathbf{x})) \geq 0$ (as shown above). Thus, $\nabla f(\mathbf{x}) = \mathbf{0}$ if $\sin(u(\mathbf{x})) = 0$ or $\nabla u(\mathbf{x}) = \mathbf{0}$.

- **Case 1:** If $\sin(u(\mathbf{x})) = 0$, then $u(\mathbf{x})$ must be a multiple of π. Since we have shown above that $u(\mathbf{x}) \in [0, \pi/4)$ (for $\mathbf{x} \in S_{\bar{r}}$) this implies $\mathbf{x} = \mathbf{0}$.
- **Case 2:** If $\nabla u(\mathbf{x}) = \mathbf{0}$, then by the definition of $u(\mathbf{x})$, this also implies that $\mathbf{x} = \mathbf{0}$.

Therefore, we conclude that

$$\nabla f(\mathbf{x}) = \mathbf{0} \iff \mathbf{x} = \mathbf{0}.$$

It follows that $\mathbf{x} = \mathbf{0}$ is a local minimum.

(f) $f(\mathbf{0}) = -1$ and $\cos : \mathbb{R} \to [-1, 1]$. From this it follows that $\mathbf{0}$ must be a global minimum of f since no element of the image of f is smaller than -1.

Solution 2:
Optimality in d dimensions

(a) $\mathsf{Var}(\mathbf{w}^\top \mathbf{X} - \mathbf{Y}) = \mathsf{Var}(\mathbf{w}^\top \mathbf{X}) + \mathsf{Var}(\mathbf{Y}) - 2\mathsf{Cov}(\mathbf{w}^\top \mathbf{X}, \mathbf{Y}) = \mathbf{w}^\top \Sigma_{\mathbf{X}} \mathbf{w} + \mathsf{Var}(\mathbf{Y}) - 2\mathbf{w}^\top \Sigma_{\mathbf{XY}}$. This is a quadratic form in $\mathbf{w}$ and $\Sigma_{\mathbf{X}}$ is p.s.d. (since it is a covariance matrix) $\Rightarrow f$ is convex.

(b) $\nabla f = 2\Sigma_{\mathbf{X}}\mathbf{w} - 2\Sigma_{\mathbf{XY}}, \nabla^2 f = 2\Sigma_{\mathbf{X}}$

(c) $\nabla f \stackrel{!}{=} \mathbf{0} \iff 2\Sigma_{\mathbf{X}}\mathbf{w} - 2\Sigma_{\mathbf{XY}} = 0 \iff \Sigma_{\mathbf{X}}\mathbf{w} = \Sigma_{\mathbf{XY}}$. This system of linear equations has a unique solution if $\Sigma_{\mathbf{X}}$ is non-singular. If $\Sigma_{\mathbf{X}}$ is non-singular it follows that $\mathbf{w} = \Sigma_{\mathbf{X}}^{-1}\Sigma_{\mathbf{XY}}$. In this case $\Sigma_{\mathbf{X}}$ is p.d. since no eigenvalue can be zero, f is strictly convex and the local minimum is global.

(d) First condition: Since $\mathbf{w}$ exists $\Sigma_{\mathbf{X}}$ must be non-singular.
Then $\Sigma_{\mathbf{X}}^{-1}\Sigma_{\mathbf{XY}} = \mathbb{E}\left((\mathbf{X}-\mathbb{E}(\mathbf{X})(\mathbf{X}-\mathbb{E}(\mathbf{X}))^\top\right)^{-1}\mathbb{E}\left((\mathbf{X}-\mathbb{E}(\mathbf{X}))(\mathbf{Y}-\mathbb{E}(\mathbf{Y}))^\top\right)$
Second condition: If $\mathbb{E}(\mathbf{X}) = \mathbf{0}, \mathbb{E}(\mathbf{Y}) = \mathbf{0}$ then
$\Sigma_{\mathbf{X}}^{-1}\Sigma_{\mathbf{XY}} = \left(\mathbb{E}(\mathbf{X}\mathbf{X}^\top)\right)^{-1}\mathbb{E}(\mathbf{X}\mathbf{Y}^\top)$.
$n(\mathbf{x}_{1:n}^\top\mathbf{x}_{1:n})^{-1}$ is a consistent estimator of $\left(\mathbb{E}(\mathbf{X}\mathbf{X}^\top)\right)^{-1}$ and
$\frac{1}{n}\mathbf{x}_{1:n}^\top y_{1:n}$ is a consistent estimator of $\mathbb{E}(\mathbf{X}\mathbf{Y}^\top)$.
$\Rightarrow$ The least squares estimator $(\mathbf{x}_{1:n}^\top\mathbf{x}_{1:n})^{-1}\mathbf{x}_{1:n}^\top y_{1:n}$ is a consistent estimator of $\left(\mathbb{E}(\mathbf{X}\mathbf{X}^\top)\right)^{-1}\mathbb{E}(\mathbf{X}\mathbf{Y}^\top)$.

Optimization Problems 1

Exercise 1: Regression

(a) Show that ridge regression is a convex problem and compute its analytical solution (given the feature matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and the target vector $\mathbf{y} \in \mathbb{R}^n$).

(b) In Bayesian regression, we are interested in the posterior density $p_{\boldsymbol{\theta}|\,\mathbf{X},\mathbf{y}}(\boldsymbol{\theta}) \propto p_{\mathbf{y}|\,\mathbf{X},\boldsymbol{\theta}}(\boldsymbol{\theta}) p_{\boldsymbol{\theta}}(\boldsymbol{\theta})$, where $p_{\mathbf{y}|\,\mathbf{X},\boldsymbol{\theta}}$ is the likelihood and $p_{\boldsymbol{\theta}}$ is the prior density. Assume the observations are i.i.d. with $y_i|\mathbf{x}_i \sim \mathcal{N}(\mathbf{x}_i^\top \boldsymbol{\theta}, 1)$ and the parameters are also i.i.d. with $\boldsymbol{\theta}_j \sim \mathcal{N}(0, \sigma_w^2)$. Find the maximizer of the posterior density. What do you observe?

(c) Find the prior density that would result in Lasso regression in b).

Exercise 2: Classification

(a) In logistic regression, we model the conditional probability $\mathbb{P}(y = 1|\mathbf{x}^{(i)}) = \frac{1}{1+\exp(-\boldsymbol{\theta}^\top \mathbf{x}^{(i)})}$ of the target $y \in \{0, 1\}$ given a feature vector $\mathbf{x}^{(i)}$. From this it follows that $\mathbb{P}(y = y^{(i)}|\mathbf{x}^{(i)}) = \mathbb{P}(y = 1|\mathbf{x}^{(i)})^{y^{(i)}}(1 - \mathbb{P}(y = 1|\mathbf{x}^{(i)})^{1-y^{(i)}}$. With this derive the empirical risk $\mathcal{R}_{\text{emp}}$ as shown in the lecture following the maximum likelihood principle. (Assume the observations are independent)

(b) Show that $\mathcal{R}_{\text{emp}}$ of a) is convex.

(c) Show that the first primal form of the linear SVM with soft constraints
$\min_{\boldsymbol{\theta},\boldsymbol{\theta}_0,\zeta^{(i)}} \frac{1}{2}\|\boldsymbol{\theta}\|_2^2 + C\sum_{i=1}^n \zeta^{(i)}$ s.t. $y^{(i)}\left(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \boldsymbol{\theta}_0\right) \geq 1 - \zeta^{(i)} \quad \forall i \in \{1, \ldots, n\}$ and $\zeta^{(i)} \geq 0 \quad \forall i \in \{1, \ldots, n\}$ and its second primal form
$\min_{\boldsymbol{\theta},\boldsymbol{\theta}_0} \sum_{i=1}^n \max(1 - y^{(i)}(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \boldsymbol{\theta}_0), 0) + \lambda\|\boldsymbol{\theta}\|_2^2$ are equivalent. What is the functional relationship between C and λ?
Hint: Try to insert the combined constraints into their associated objective.

(d) Show that the second primal form of the linear SVM is a convex problem

Optimization Problems 1

Solution 1: Regression

(a) Let $f : \mathbb{R}^d \to \mathbb{R}, \boldsymbol{\theta} \mapsto 0.5\|\mathbf{X}\boldsymbol{\theta} - \mathbf{y}\|_2^2 + 0.5 \cdot \lambda\|\boldsymbol{\theta}\|_2^2, \lambda > 0$
$\frac{\partial}{\partial \boldsymbol{\theta}} f = \boldsymbol{\theta}^\top \mathbf{X}^\top \mathbf{X} - \mathbf{y}^\top \mathbf{X} + \lambda \boldsymbol{\theta}^\top \overset{!}{=} \mathbf{0} \iff \boldsymbol{\theta}^\top (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}) = \mathbf{y}^\top \mathbf{X}$
$\Rightarrow \boldsymbol{\theta} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}$.
$\frac{\partial^2}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^\top} f = \underbrace{\mathbf{X}^\top \mathbf{X}}_{\text{p.s.d.}} + \underbrace{\lambda \mathbf{I}}_{\text{p.d. if } \lambda > 0}$ is p.d. if $\lambda > 0 \Rightarrow f$ is (strictly) convex

(b) Since the observations and parameters are assumed to be i.i.d. it follows that
$p_{\boldsymbol{\theta}|\, \mathbf{X}, \mathbf{y}}(\boldsymbol{\theta}) \propto p_{\mathbf{y}|\, \mathbf{X}, \boldsymbol{\theta}}(\boldsymbol{\theta}) p_{\boldsymbol{\theta}}(\boldsymbol{\theta}) \propto \exp\left(-\frac{(\mathbf{X}\boldsymbol{\theta} - \mathbf{y})^\top \mathbf{I}^{-1} (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})}{2}\right) \exp\left(-\frac{\boldsymbol{\theta}^\top \boldsymbol{\theta}}{2\sigma_w^2}\right)$.
The minimizer of the negative log posterior density is maximizer of posterior density and hence
$\boldsymbol{\theta}^* = \arg\min_\theta - \log\left(\exp\left(-\frac{(\mathbf{X}\boldsymbol{\theta} - \mathbf{y})^\top (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})}{2} - \frac{\boldsymbol{\theta}^\top \boldsymbol{\theta}}{2\sigma_w^2}\right)\right) = \arg\min_\theta \frac{1}{2}\|\mathbf{X}\boldsymbol{\theta} - \mathbf{y}\|_2^2 + \frac{1}{2 \cdot \sigma_w^2}\|\boldsymbol{\theta}\|_2^2$.
This is ridge regression and the solution follows from a) with $\lambda = 1/\sigma_w^2$.

(c) From b) we see that for the density of interest it must hold that
$-\log p(\theta) = 0.5 \cdot \lambda|\theta| + c$ with $c \in \mathbb{R} \iff p(\theta) \propto \exp(-0.5 \cdot \lambda|\theta|)$.
$\Rightarrow \theta \overset{\text{i.i.d.}}{\sim} \text{Laplace}(0, 2/\lambda)$.

Solution 2: Classification

(a) First observe that $1 - \mathbb{P}(y = 1|\mathbf{x}^{(i)}) = \frac{\exp(-\boldsymbol{\theta}^\top \mathbf{x}^{(i)})}{1 + \exp(-\boldsymbol{\theta}^\top \mathbf{x}^{(i)})} = \frac{1}{1 + \exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)})} = \mathbb{P}(y = 1| - \mathbf{x}^{(i)})$.
Define $\sigma(\mathbf{x}) := \mathbb{P}(y = 1|\mathbf{x}^{(i)})$.
With this we get that $\log\left(\mathbb{P}(y = y^{(i)}|\mathbf{x}^{(i)})\right) = \log\left(\mathbb{P}(y = 1|\mathbf{x}^{(i)})^{y^{(i)}} (1 - \mathbb{P}(y = 1|\mathbf{x}^{(i)})^{1 - y^{(i)}}\right)$
$= y^{(i)} \log(\sigma(\mathbf{x}^{(i)})) + (1 - y^{(i)}) \log(1 - \sigma(\mathbf{x}^{(i)}))$
$= y^{(i)} (\log(\sigma(\mathbf{x}^{(i)}) - \log(\sigma(-\mathbf{x}^{(i)}))) + \log(\sigma(-\mathbf{x}^{(i)}))$
$= y^{(i)} \left(\log\left(\frac{\sigma(\mathbf{x}^{(i)})}{\sigma(-\mathbf{x}^{(i)})}\right)\right) + \log(\sigma(-\mathbf{x}^{(i)}))$
$= y^{(i)} \left(\log\left(\frac{1 + \exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)})}{1 + \exp(-\boldsymbol{\theta}^\top \mathbf{x}^{(i)})}\right)\right) - \log(1 + \exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)}))$
$= y^{(i)} \left(\log\left(\exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)}) \frac{1 + \exp(-\boldsymbol{\theta}^\top \mathbf{x}^{(i)})}{1 + \exp(-\boldsymbol{\theta}^\top \mathbf{x}^{(i)})}\right)\right) - \log(1 + \exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)}))$
$= y^{(i)} \boldsymbol{\theta}^\top \mathbf{x}^{(i)} - \log(1 + \exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)}))$
With this we find that $\mathcal{R}_{\text{emp}} = -\log \prod_{i=1}^n \mathbb{P}(y = y^{(i)}|\mathbf{x}^{(i)}) = \sum_{i=1}^n \log(1 + \exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)})) - y^{(i)} \boldsymbol{\theta}^\top \mathbf{x}^{(i)}$

(b) $\frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{R}_{\text{emp}} = \sum_{i=1}^n \frac{\exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)})}{1 + \exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)})} \mathbf{x}^{(i)\top} - y^{(i)} \mathbf{x}^{(i)\top}$
$\frac{\partial^2}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^\top} \mathcal{R}_{\text{emp}} = \sum_{i=1}^n \frac{\exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)})(1 + \exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)}) - \exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)})^2}{(1 + \exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)})^2} \mathbf{x}^{(i)} \mathbf{x}^{(i)\top} = \sum_{i=1}^n \underbrace{\frac{\exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)})}{(1 + \exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)})^2}}_{>0} \underbrace{\mathbf{x}^{(i)} \mathbf{x}^{(i)\top}}_{\text{p.s.d.}}$
is p.s.d. $\Rightarrow \mathcal{R}_{\text{emp}}$ is convex.

(c) We can transform the inequalities such that

$$\zeta^{(i)} \geq 1 - y^{(i)} \left(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \boldsymbol{\theta}_0\right) \quad \text{and} \quad \zeta^{(i)} \geq 0$$

for all $i \in \{1, \ldots, n\}$. However, for a minimizer of the first primal form, it has to hold that

$$\zeta^{(i)} = \begin{Bmatrix} 1 - y^{(i)} \left(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \boldsymbol{\theta}_0\right) & \text{if } 1 - y^{(i)} \left(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \boldsymbol{\theta}_0\right) \geq 0 \\ 0 & \text{if } 1 - y^{(i)} \left(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \boldsymbol{\theta}_0\right) < 0 \end{Bmatrix} = \max(1 - y^{(i)} \left(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \boldsymbol{\theta}_0\right), 0),$$

since, otherwise, it would not be a minimizer.

Now, we can insert $\zeta^{(i)}$ into the objective function and get

$$f(\boldsymbol{\theta}) = \frac{1}{2}\|\boldsymbol{\theta}\|_2^2 + C\sum_{i=1}^{n} \max(1 - y^{(i)}\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \boldsymbol{\theta}_0, 0).$$

Minimizing f is equivalent to minimizing f/C, i.e.,

$$\sum_{i=1}^{n} \max(1 - y^{(i)}(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \boldsymbol{\theta}_0), 0) + \lambda\|\boldsymbol{\theta}\|_2^2$$

for $\lambda = 1/(2C)$.

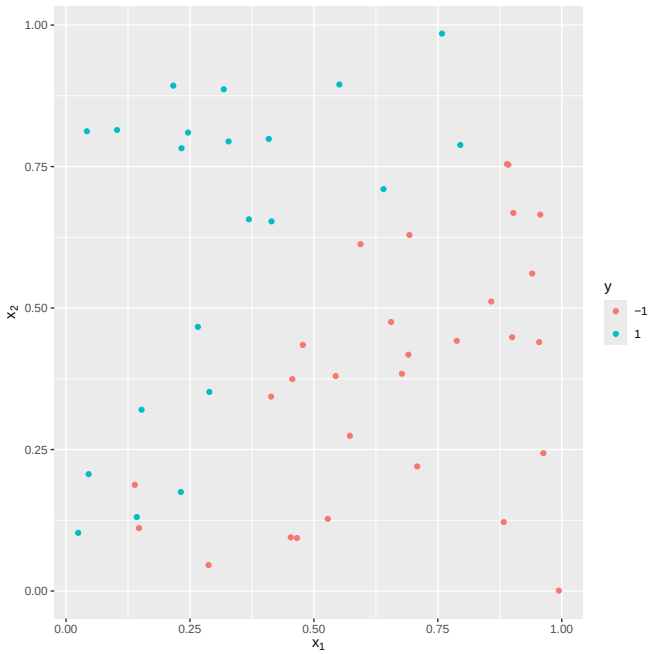
(d) First we show that $g : \mathbb{R} \to \mathbb{R}, x \mapsto \max(x, 0)$ is convex:
$g(x) = 0.5|x| + 0.5x \Rightarrow \max(x, 0)$ is convex since it is the sum of two convex functions.
Also g is increasing $\Rightarrow \max(1 - y^{(i)}\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \boldsymbol{\theta}_0, 0)$ is convex since $1 - y^{(i)}\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \boldsymbol{\theta}_0$ is convex (linear).
With this we can conclude that $\sum_{i=1}^{n} \max(1 - y^{(i)}(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \boldsymbol{\theta}_0), 0) + \lambda\|\boldsymbol{\theta}\|_2^2$ is convex since it is the sum of convex functions.

Univariate Optimization 1

Exercise 1: Golden Ratio, Brent's Method

You are given the following data situation:

```
library(ggplot2)

set.seed(123)

X = matrix(runif(100), ncol = 2)
y = -((X %*% c(-1, 1) + rnorm(50, 0, 0.1) < 0) * 2 - 1)
df = as.data.frame(X)
df$type = as.character(y)

ggplot(df) +
  geom_point(aes(x = V1, y = V2, color=type)) +
  xlab(expression(x[1])) +
  ylab(expression(x[2])) +
  labs(color="y")
```



In the following we want to estimate a linear SVM without intercept and with $\lambda = 1$. We assume we know that $\boldsymbol{\theta}_2 = 2$.

(a) Show that if $f : \mathbb{R}^2 \to \mathbb{R}$ is convex then $g_c : \mathbb{R} \to \mathbb{R}, x \mapsto f(x, c) \quad \forall c \in \mathbb{R}$ is convex.

(b) Explain why the non-geometric primal linear SVM formulation should be used rather than the geometric one if we want to find $\boldsymbol{\theta}_1$ via the golden ratio algorithm[1].

(c) Find $\boldsymbol{\theta}_1$ via the golden ratio algorithm. Implement the algorithm in `R`. For the termination criterion, use an absolute error of 0.01. Use $[-3, 3]$ as the starting interval.

[1] We choose this algorithm for educational purposes; in practice, we typically use more advanced algorithms.

(d) Given the three points $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ show that the parameters $a, b, c \in \mathbb{R}$ of the interpolating parabola can be found via $\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ x_3^2 & x_3 & 1 \end{pmatrix}^{-1} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$ when the parabola equation is given by $p(x) = ax^2 + bx + c$.

(e) Find $\boldsymbol{\theta}_1$ via Brent's method[2]. Implement a simplified version[3] of the algorithm in `R`. For the termination criterion, use an absolute error of 0.01. Use $[-3, 3]$ as the starting interval. For the first step, use a golden ratio step.

(f) Now, assume we do not know $\boldsymbol{\theta}_2$. Our initial guess is $\boldsymbol{\theta}_2 = 0$. We now alternately minimize w.r.t. either $\boldsymbol{\theta}_1$ or $\boldsymbol{\theta}_2$ via the golden ratio method (the starting interval is always reset to $[-3, 3]$) while the other parameter is held constant. We switch to minimizing the other parameter when the absolute error is smaller than 0.01. Repeat this procedure 10 times.

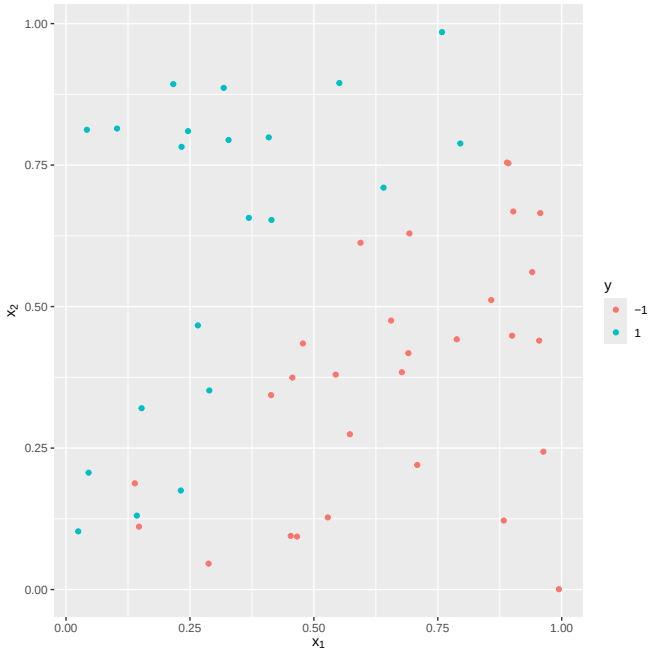
(g) How does the optimization trace of f) look in parameter space?

[2]We choose this algorithm for educational purposes; in practice, we typically use more advanced algorithms.
[3]Only check if the proposed point is in the current interval

Univariate Optimization 1

Solution 1: Golden Ratio, Brent's Method

```
library(ggplot2)

set.seed(123)

X = matrix(runif(100), ncol = 2)
y = -((X %*% c(-1, 1) + rnorm(50, 0, 0.1) < 0) * 2 - 1)
df = as.data.frame(X)
df$type = as.character(y)

ggplot(df) +
  geom_point(aes(x = V1, y = V2, color=type)) +
  xlab(expression(x[1])) +
  ylab(expression(x[2])) +
  labs(color="y")
```



(a) Since f is convex it holds for arbitrary $\mathbf{x}, \mathbf{y} \in \mathbb{R}^2, t \in [0,1]$ that
$f(\mathbf{x} + t(\mathbf{y} - \mathbf{x})) \leq f(\mathbf{x}) + t(f(\mathbf{y}) - f(\mathbf{x}))$.
This means this holds also for $\mathbf{x}_c = (x, c)^\top$ and $\mathbf{y}_c = (y, c)^\top$ with $x, y \in \mathbb{R}$ and fixed $c \in \mathbb{R}$:
$f(\mathbf{x}_c + t(\mathbf{y}_c - \mathbf{x}_c)) \leq f(\mathbf{x}_c) + t(f(\mathbf{y}_c) - f(\mathbf{x}_c))) \iff g_c(x + t(y - x)) \leq g_c(x) + t(g_c(y) - g_c(x))$.
$\Rightarrow g_c$ is convex.

(b) The non-geometric primal linear SVM formulation is convex and unconstrained $\Rightarrow$ For one parameter the objective is also convex (a) and we can directly use GR. In contrast, the geometric formulation has linear constraints.

(c)
```r
# Define objective
f <- function(theta) theta %*% theta +
    sum(sapply(1 - y * (X %*% theta), function(x) max(x, 0)))

# Objective w.r.t theta_1 with fixed theta_2
ft1 <- function(theta_1) f(c(theta_1, 2))

phi = (sqrt(5) - 1)/2

gr <- function(f, lx=-3, rx=3, abs_error = 0.01){

  # initialize variables needed for stopping criterion
  fbest_old = Inf
  xbest_old = Inf
  xbest = NA

  # compute candidate xs
  dist = rx - lx
  cx = c(lx + (1-phi) * dist, rx - (1-phi) * dist)

  while(TRUE){
    fcx1 = f(cx[1])
    fcx2 = f(cx[2])

    # check which candidate is better and update cx
    if (fcx1 < fcx2){
      fbest = fcx1
      xbest = cx[1]
      rx = cx[2]
      cx[2] = cx[2] - (cx[1] - lx)
    }else{
      fbest = fcx2
      xbest = cx[2]
      lx = cx[1]
      cx[1] = cx[1] + (rx - cx[2])
    }
    # assure cx[1] < cx[2]
    cx = sort(cx)

    # check if we need to stop the loop depending on the termination criterion
    if (abs(xbest_old - xbest) < abs_error){
      return(c(xbest, fbest))
    }
    fbest_old = fbest
    xbest_old = xbest
  }
}

gr(ft1)

## [1] -2.45898 29.91294
```

(d) We are given three equations:
$ax_1^2 + bx_1 + c = y_1$
$ax_2^2 + bx_2 + c = y_2$

$ax_3^2 + bx_3 + c = y_3$. Which we can express equivalently as $\underbrace{\begin{pmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ x_3^2 & x_3 & 1 \end{pmatrix}}_{:=\mathbf{\Lambda}} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$. The result follows straightforwardly assuming $\mathbf{\Lambda}$ is non-singular.

(e)
```
gr_step <- function(f, lx, rx){
  dist = rx - lx

  # compute candidates
  cxs = c(lx + (1-phi) * dist,
    rx - (1-phi) * dist)

  fcx = sapply(cxs, f)

  # find best candidate
  if (fcx[1] < fcx[2]){
    fbest = fcx[1]
    cx = cxs[1]
    rx = cxs[2]
  }else{
    fbest = fcx[2]
    cx = cxs[2]
    lx = cxs[1]
  }

  return(c(lx, rx, cx, fbest))
}

brent <- function(f, lx = -3, rx = 3, abs_error = 0.01){

  fbest_old = Inf
  xbest = NA
  xbest_old = Inf

  # we do not have a valid candidate in the beginning
  cx = Inf

  while(TRUE){
    # if candidate is not valid do a golden ratio step
    if(cx <= lx | cx >= rx){
      res = gr_step(f, lx, rx)
      lx = res[1]
      rx = res[2]
      cx = res[3]
      xbest = cx
      fbest = res[4]
    }else{ # try doing quadratic interpolation otherwise
      # compute objective values
      xs = c(lx, rx, cx)
      fxs = sapply(xs, f)

      # find parameters of the interpolating parabola
      params = solve(t(sapply(xs, function(x) c(x^2, x, 1))), fxs)
      # find minimum of the parabola
      cx_new = -params[2]/(2*params[1])

      # if candidate is valid do quadratic interpolation step
      if(cx_new < rx & cx_new > lx){
```

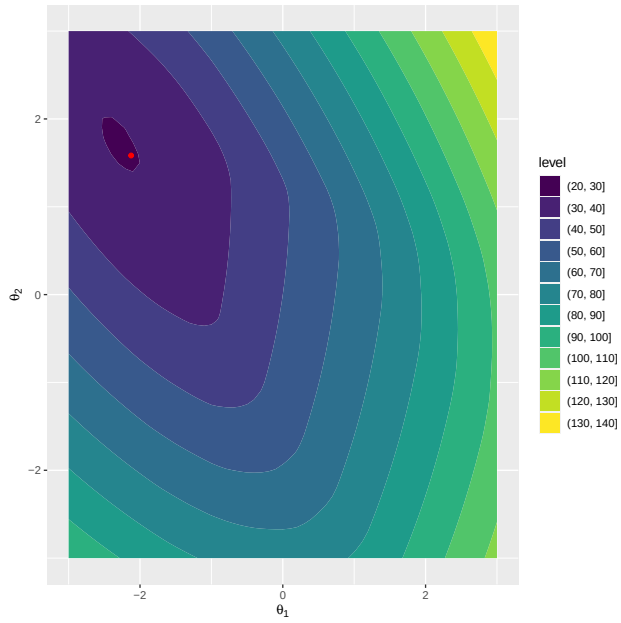
```
        cxs = sort(c(cx, cx_new))
        fcx = sapply(cxs, f)

        # find best candidate
        if (fcx[1] < fcx[2]){
          fbest = fcx[1]
          cx = cxs[1]
          rx = cxs[2]
        }else{
          fbest = fcx[2]
          cx = cxs[2]
          lx = cxs[1]
        }
        xbest = cx

      }
    }

  # check if we need to stop the loop depending on the termination criterion
  if (abs(xbest - xbest_old) < abs_error){
    return(c(xbest, fbest))
  }
  fbest_old = fbest
  xbest_old = xbest
  }
}

brent(ft1)

## [1] -2.409456 29.903281
```

(f)

```
# intialize thetas
t1 = 0
t2 = 0

for(i in 0:9){
  # alternate between univariately optimizing each parameter while the other
  # is fixed
  if(i %% 2 == 0){
    ft <- function(t) f(c(t, t2))
    res = gr(ft)
    t1 = res[1]

  }else{
    ft <- function(t) f(c(t1, t))
    res = gr(ft)
    t2 = res[1]
  }
  print(c(t1, t2, f(c(t1, t2))))
}

## [1] -1.583592  0.000000 37.878640
## [1] -1.583592  1.583592 32.490253
## [1] -2.124612  1.583592 29.742862
## [1] -2.124612  1.583592 29.742862
## [1] -2.124612  1.583592 29.742862
## [1] -2.124612  1.583592 29.742862
```

```
## [1] -2.124612  1.583592 29.742862
## [1] -2.124612  1.583592 29.742862
## [1] -2.124612  1.583592 29.742862
## [1] -2.124612  1.583592 29.742862
```

```
x = seq(-3, 3, by=0.1)
xx = expand.grid(X1 = x, X2 = x)

fxx = apply(xx, 1, f)
df = data.frame(xx = xx, fxx = fxx)

ggplot() +
    geom_contour_filled(data = df, aes(x = xx.X1, y = xx.X2, z = fxx)) +
    xlab(expression(theta[1])) +
    ylab(expression(theta[2])) +
    geom_point(data = as.data.frame(t(c(t1, t2))), mapping = aes(x=V1, y=V2),
               color="red")
```



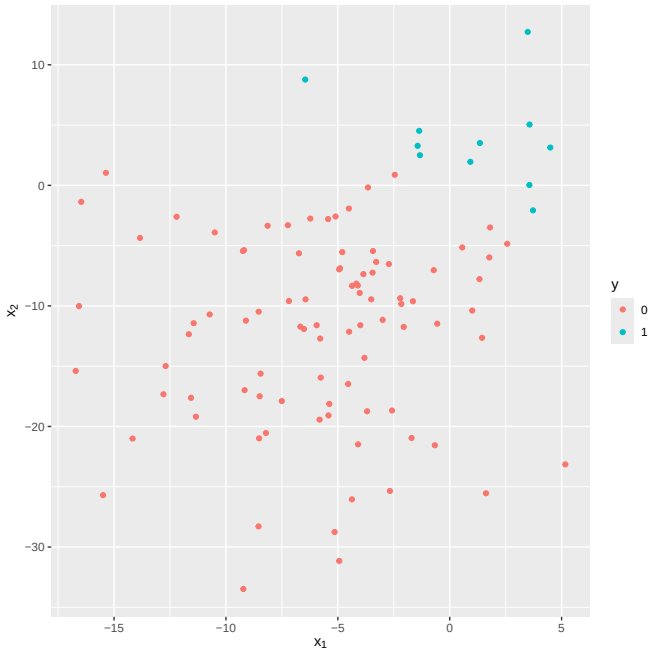
(g) The trace looks like a orthogonal zig-zag line.

Multivariate Optimization 1

Exercise 1: Gradient Descent

You are given the following data situation:

```
library(ggplot2)

set.seed(314)
n <- 100
X = cbind(rnorm(n, -5, 5),
  rnorm(n, -10, 10))
X_design = cbind(1, X)

z <- 2*X[,1] + 3*X[,2]
pr <- 1/(1+exp(-z))
y <- as.integer(pr > 0.5)
df <- data.frame(X = X, y = y)

ggplot(df) +
  geom_point(aes(x = X.1, y = X.2, color=y)) +
  xlab(expression(x[1])) +
  ylab(expression(x[2]))
```



In the following we want to estimate a logistic regression without intercept via gradient descent[1].

(a) The data situation is called complete separation, i.e., the classes can be perfectly classified with a linear classifier. Show that in this situation if $\tilde{\boldsymbol{\theta}}$ perfectly classifies the data then:
$\mathcal{R}_{\text{emp}}(\tilde{\boldsymbol{\theta}}) > \mathcal{R}_{\text{emp}}(\alpha\tilde{\boldsymbol{\theta}})$ with $\alpha > 1$.

(b) Visualize $\mathcal{R}_{\text{emp}}$ in $[-1, 4] \times [-1, 4]$.

[1] We chose this algorithm for educational purposes; in practice, we typically use second order algorithms.

(c) Find the gradient of $\mathcal{R}_{\text{emp}}$ for arbitrary $\boldsymbol{\theta}$.

(d) Solve the logistic regression via gradient descent. Use a step size $\alpha = 0.01$, starting point $\boldsymbol{\theta}^{[0]} = (0, 0)^\top$ and train for 500 steps. Repeat this with $\alpha = 0.02$. Explain your observation.
Hint: a)

(e) Repeat d) but add an L2 penalization term (with $\lambda = 1$) to the objective. What do you observe now?

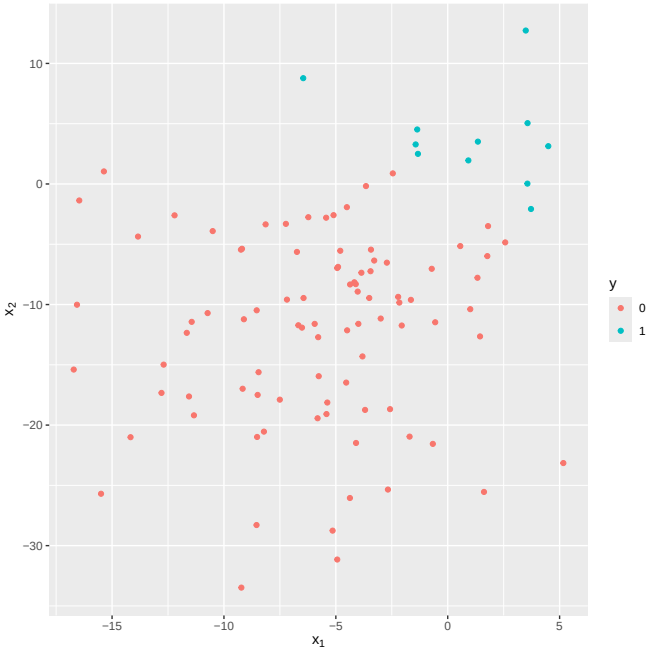
(f) Visualize the regularized $\mathcal{R}_{\text{emp}}$ in $[-1, 4] \times [-1, 4]$.

(g) Repeat e) but with backtracking. Set $\gamma = 0.9$ and $\tau = 0.5$

Multivariate Optimization 1

Solution 1: Gradient Descent
You are given the following data situation:

```
library(ggplot2)

set.seed(314)
n <- 100
X = cbind(rnorm(n, -5, 5),
  rnorm(n, -10, 10))
X_design = cbind(1, X)

z <- 2*X[,1] + 3*X[,2]
pr <- 1/(1+exp(-z))
y <- as.integer(pr > 0.5)
df <- data.frame(X = X, y = y)

ggplot(df) +
  geom_point(aes(x = X.1, y = X.2, color=as.factor(y))) +
  xlab(expression(x[1])) +
  ylab(expression(x[2])) +
  labs(colour = "y")
```



(a) We start with

$$\begin{aligned}\mathcal{R}_{\text{emp}}(\tilde{\boldsymbol{\theta}}) &= \sum_{i=1}^{n} \log(1+\exp(\tilde{\boldsymbol{\theta}}^\top \mathbf{x}^{(i)})) - y^{(i)} \tilde{\boldsymbol{\theta}}^\top \mathbf{x}^{(i)} \\ &= \sum_{i=1}^{n} \begin{cases} \log(1+\exp(\tilde{\boldsymbol{\theta}}^\top \mathbf{x}^{(i)})) & \text{if } y^{(i)} = 0, \\ \log(1+\exp(\tilde{\boldsymbol{\theta}}^\top \mathbf{x}^{(i)})) - \tilde{\boldsymbol{\theta}}^\top \mathbf{x}^{(i)} & \text{if } y^{(i)} = 1. \end{cases}\end{aligned}$$

Since $\tilde{\boldsymbol{\theta}}$ perfectly classifies the data, we know that

$$\begin{cases} \tilde{\boldsymbol{\theta}}^\top \mathbf{x}^{(i)} < 0 & \text{if } y^{(i)} = 0, \\ \tilde{\boldsymbol{\theta}}^\top \mathbf{x}^{(i)} \geq 0 & \text{if } y^{(i)} = 1. \end{cases}$$

Hence, we can focus on the functions $g(z) = \log(1 + \exp(-z))$ and $h(z) = \log(1 + \exp(z)) - z$ for $z > 0$ and study their monotonicity.

We compute

$$g'(z) = \frac{1}{1 + \exp(-z)} \cdot \exp(-z) \cdot (-1) = -\underbrace{\frac{\exp(-z)}{1 + \exp(-z)}}_{>0} < 0$$

and

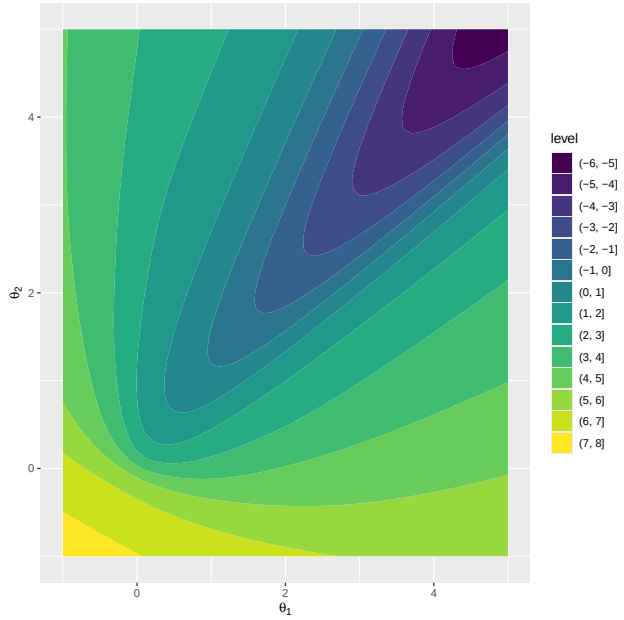
$$h'(z) = \underbrace{\frac{\exp(z)}{1 + \exp(z)}}_{<1} - 1 < 0.$$

Therefore, both g and h are strictly monotonically decreasing.

It follows that $\mathcal{R}_{\text{emp}}(\tilde{\boldsymbol{\theta}}) > \mathcal{R}_{\text{emp}}(\alpha\tilde{\boldsymbol{\theta}})$ for $\alpha > 1$.

(b)

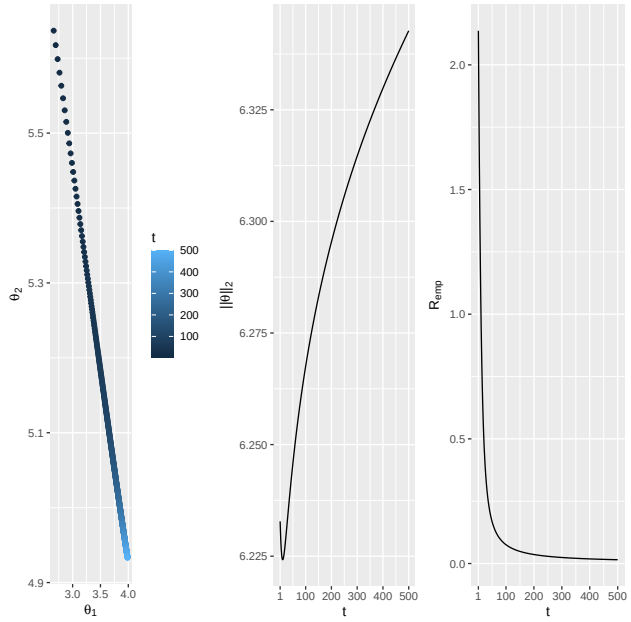
```
lambda = 0

f <- function(theta, lambda) lambda * theta %*% theta +
  sum(-y * X %*% theta + log(1 + exp(X %*% theta)))

x = seq(-1, 5, by=0.1)
xx = expand.grid(X1 = x, X2 = x)

fxx = log(apply(xx, 1, function(t) f(t, lambda)))
df = data.frame(xx = xx, fxx = fxx)

ggplot() +
    geom_contour_filled(data = df, aes(x = xx.X1, y = xx.X2, z = fxx)) +
    xlab(expression(theta[1])) +
    ylab(expression(theta[2]))
```



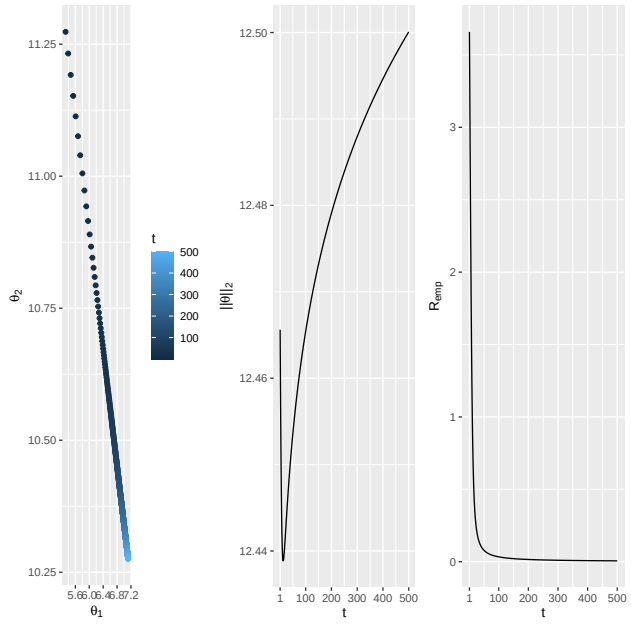
(c) $\frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{R}_{\text{emp}} = \sum_{i=1}^{n} \frac{\exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)})}{1+\exp(\boldsymbol{\theta}^\top \mathbf{x}^{(i)})} \mathbf{x}^{(i)\top} - y^{(i)} \mathbf{x}^{(i)\top}$

(d) Note that we visualize form the first iteration on ($t = 1$) and not from the initial starting point $\boldsymbol{\theta}^{[0]} = (0, 0)^\top$ but after having already made one GD step.

```
library(gridExtra)

plot_fun <- function(gd_fun, lambda){
  theta = c(0,0)
  thetas = NULL
  thetas_norm = NULL
  fs = NULL
  for(i in 1:500){
    theta = gd_fun(theta)
    thetas_norm = rbind(thetas_norm, sqrt(theta %*% theta))
    thetas = rbind(thetas, t(theta))
    fs = rbind(fs, f(theta, lambda))
  }

  df_trace = as.data.frame(thetas)
  df_trace$t = 1:nrow(df_trace)
  trace_plot = ggplot() +
    geom_point(data = df_trace, aes(x=V1, y=V2, colour=t)) +
    xlab(expression(theta[1])) +
    ylab(expression(theta[2]))
  norm_plot = ggplot(data.frame(norms = thetas_norm, t = 1:nrow(thetas_norm))) +
    geom_line(aes(x = t, y = norms)) +
    scale_x_continuous(breaks = c(1, 100, 200, 300, 400, 500), limits = c(1, 500)) +
    ylab(expression(paste("||", theta, "||"[2])))
  remp_plot = ggplot(data.frame(f = fs, t = 1:nrow(thetas_norm))) +
    geom_line(aes(x = t, y = f)) +
    scale_x_continuous(breaks = c(1, 100, 200, 300, 400, 500), limits = c(1, 500)) +
    ylab(expression(R[emp]))
  grid.arrange(trace_plot, norm_plot, remp_plot, ncol=3)
}

df_t <- function(theta, lambda) lambda * t(theta) -(t(y) %*% X) +
  t(1/(1 + exp(-X %*% theta))) %*% X

gd_step <- function(theta, alpha, lambda) return(theta - alpha * df_t(theta, lambda)[1,])

## Alpha = 0.01
gd_fun <- function(theta) return(gd_step(theta, 0.01, lambda))
plot_fun(gd_fun, 0)
```



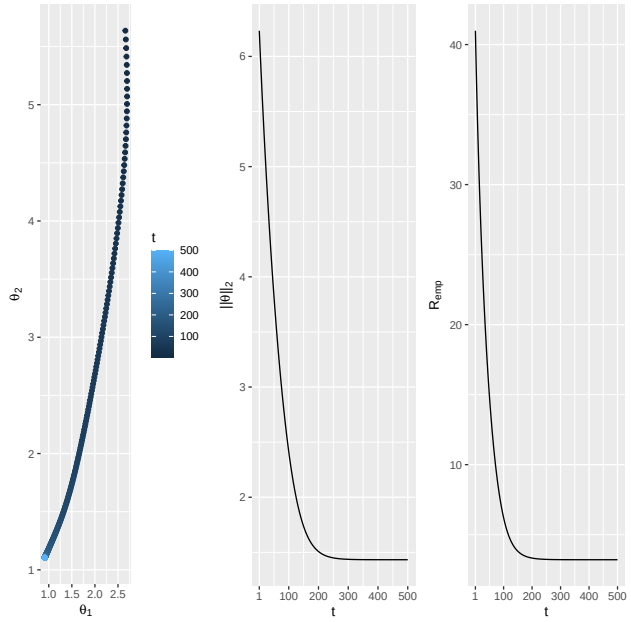

```
## Alpha = 0.02
gd_fun <- function(theta) return(gd_step(theta, 0.02, lambda))
plot_fun(gd_fun, 0)
```



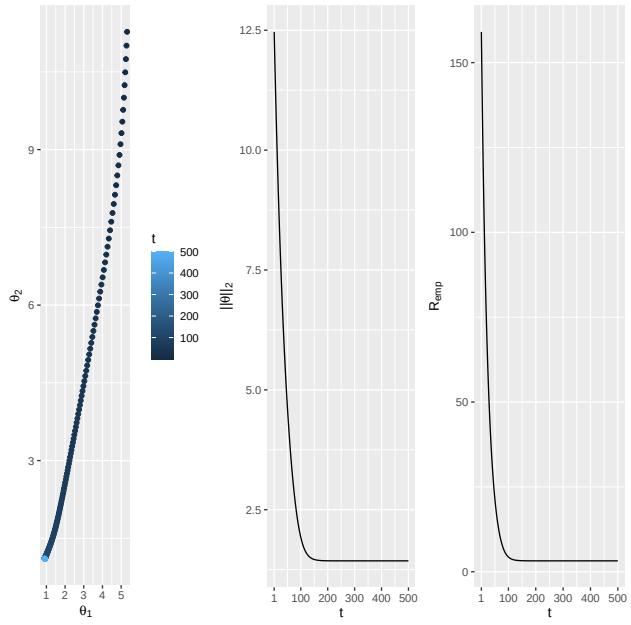
Gradient descent will in theory not converge since $\mathcal{R}_{\text{emp}}$ has no minimum (a)

(e)

```
## Lambda = 1, alpha = 0.01
gd_fun <- function(theta) return(gd_step(theta, 0.01, 1))
plot_fun(gd_fun, 1)
```

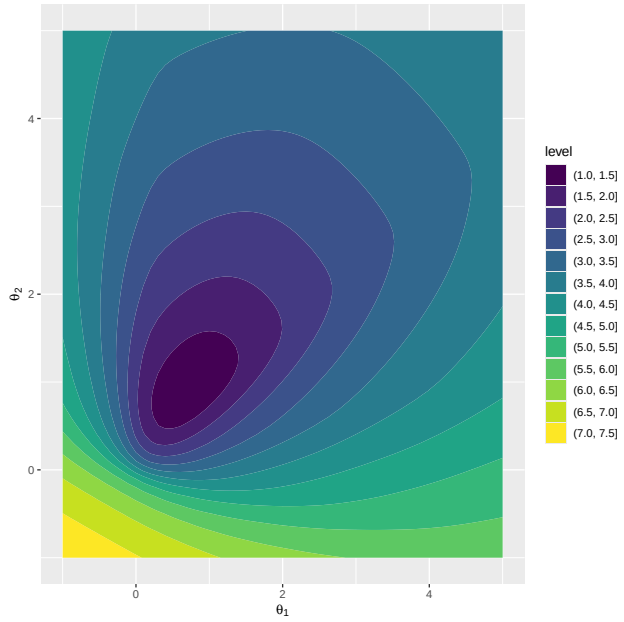



```
## Lambda = 1, alpha = 0.02
gd_fun <- function(theta) return(gd_step(theta, 0.02, 1))
plot_fun(gd_fun, 1)
```



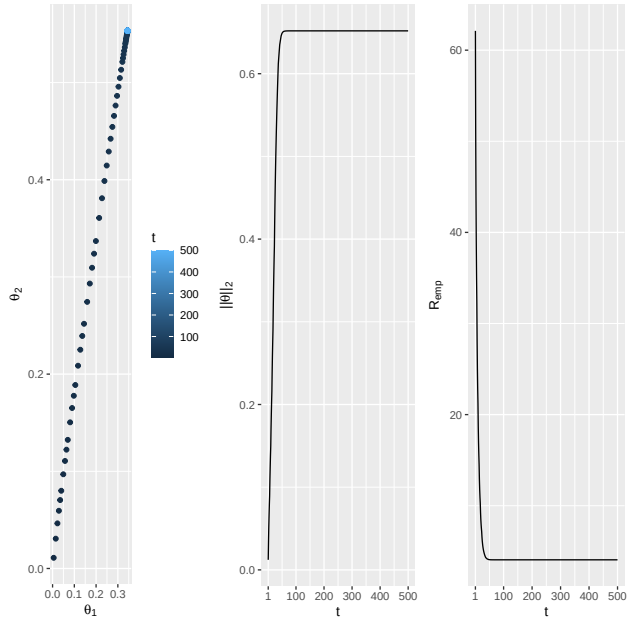
(f)

```
lambda = 1

fxx_reg = log(apply(xx, 1, function(t) f(t, lambda)))
df_reg = data.frame(xx = xx, fxx = fxx_reg)

ggplot() +
    geom_contour_filled(data = df_reg, aes(x = xx.X1, y = xx.X2, z = fxx)) +
    xlab(expression(theta[1])) +
    ylab(expression(theta[2]))
```

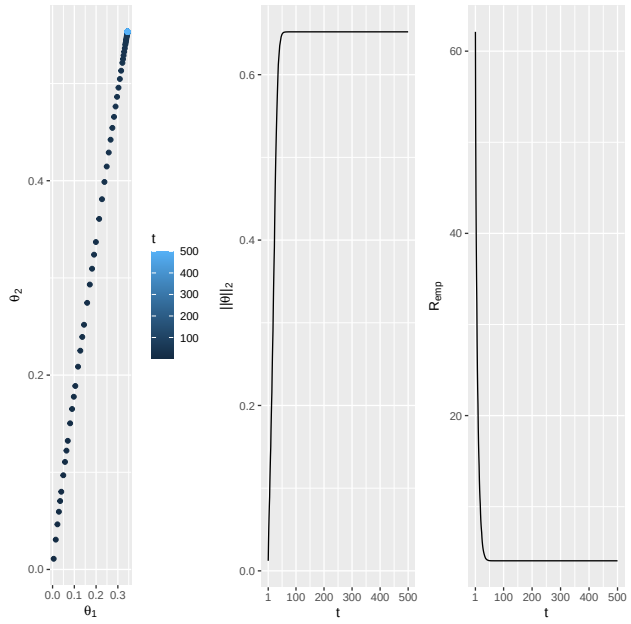


(g)

```r
gd_backtracking_step <- function(theta, alpha, gamma, tau, lambda){
    ftheta = f(theta, lambda)
    dftheta = df_t(theta, lambda)[1,]
    for(i in 1:1000) {
      theta_prop = theta - alpha * dftheta
      if(f(theta_prop, lambda) <= ftheta - gamma * alpha * t(dftheta) %*% dftheta){
        return(theta_prop)
      }else{
        alpha = tau * alpha
      }
    }
    return(theta)
}

## Lambda = 1, alpha = 0.01
gd_fun <- function(theta) return(gd_backtracking_step(theta, 0.01, 0.9, 0.5, 1))
plot_fun(gd_fun, 1)
```



```r
## Lambda = 1, alpha = 0.02
gd_fun <- function(theta) return(gd_backtracking_step(theta, 0.02, 0.9, 0.5, 1))
plot_fun(gd_fun, 1)
```



Multivariate Optimization 2

Exercise 1: Gradient Descent

A radial basis function (RBF) network has been fitted to a unknown blackbox function resulting in a model $f : \mathbb{R}^2 \to \mathbb{R}, \mathbf{x} \mapsto \sum_{i=1}^{2} w_i \cdot \rho(\|\mathbf{x} - \mathbf{c}_i\|_{S_i})$
with $c_1 = (-1.1, 1.1)^\top, c_2 = (0.8, -0.8)^\top$, quartic (biweight) kernel function
$\rho : \mathbb{R} \to \mathbb{R}, u \mapsto \begin{cases} (1-u^2)^2 & |u| < 1 \\ 0 & \text{otherwise} \end{cases}$, $w_1 = 1, w_2 = -1$ and Mahalanobis distance $\|\cdot\|_{S_i}$ with covariance matrices
$S_1 = \mathbf{I}$ and $S_2 = \begin{pmatrix} 1.1 & -0.9 \\ -0.9 & 1.1 \end{pmatrix}$.
The Mahalanobis distance is given by $\|\mathbf{x} - \mathbf{c}\|_S = \sqrt{(\mathbf{x} - \mathbf{c})^\top S^{-1} (\mathbf{x} - \mathbf{c})}$.
(Note: We chose the kernel function and the distance measure for educational purposes; often, a Gaussian kernel and the Euclidean distance are used in practice.)

(a) Plot f in the range $[-2, 2] \times [-2, 2]$

(b) Show that $\cap_{i=1}^{2} \{\mathbf{x} \in \mathbb{R}^2 |\ \rho(\|\mathbf{x} - \mathbf{c}_i\|_{S_i}) \neq 0\} = \emptyset$.

(c) Find the global minimum of f analytically.
Hint: b)

(d) Write an `R` script which computes two gradient descent steps starting at $x^{[0]} = (-0.45, 0.5)^\top$ with step size $\alpha = 0.15$. What do you observe?

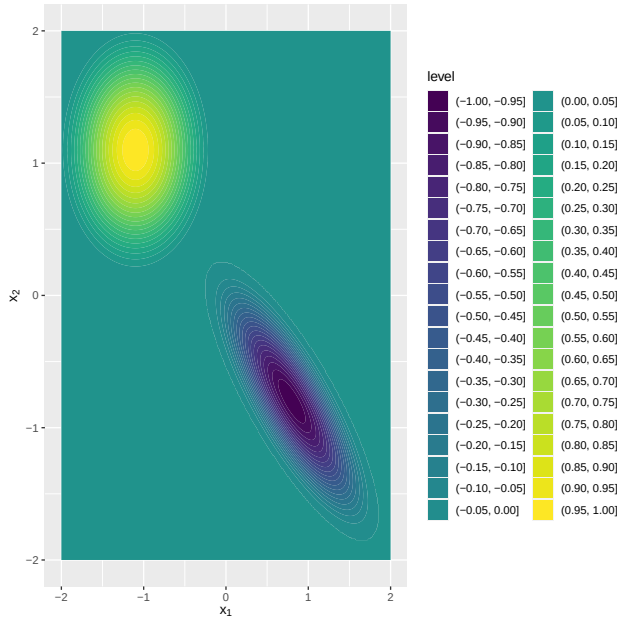
(e) Perform analytically two gradient descent steps starting at $x^{[0]} = (-0.45, 0.5)^\top$ with step size $\alpha = 0.15$.

(f) Write an `R` script which finds the global minimum with the settings in e) but with momentum. (Set $\nu^{[0]} = (0.4, -0.4)^\top, \varphi = 0.5$ and stop after 15 iterations.)

Multivariate Optimization 2

Solution 1: Gradient Descent

(a)
```
library(ggplot2)

c1 = c(-1.1,1.1)
c2 = c(0.8, -0.8)

S2 = matrix(c(1.1, -0.9, -0.9, 1.1), nrow = 2)
S2_inv = solve(S2)

rho <- function(u) {ifelse(abs(u) < 1, (1 - u^2)^2, 0)}

dist1 <- function(x) {sqrt((x - c1) %*% (x -c1))}
dist2 <- function(x) {sqrt((x - c2)  %*% S2_inv %*% (x - c2))}

f <- function(x) {rho(dist1(x)) - rho(dist2(x))}

x = seq(-2, 2, by=0.01)
xx = expand.grid(X1 = x, X2 = x)

fxx = apply(xx, 1, f)
df = data.frame(xx = xx, fxx = fxx)

cont_plot = ggplot() +
    geom_contour_filled(data = df, aes(x = xx.X1, y = xx.X2, z = fxx),
                        binwidth = 0.05) +
  xlab(expression(x[1])) +
  ylab(expression(x[2]))
cont_plot
```



(b) First we analyze $\rho(u)$ for $|u| < 1 : (1-u^2)^2 = 0 \iff (1-u^2) = 0 \iff u^2 = 1 \Rightarrow \rho(u) \neq 0$ for $u^2 < 1$ and $\rho(u) = 0$ for $u^2 \geq 1$.
We can check this condition for both squared distances around the centers $\mathbf{c}_1, \mathbf{c}_2$:

(i) $\|\mathbf{x} - \mathbf{c}_1\|_{S_1}^2 < 1 \iff \|\mathbf{x} - \mathbf{c}_1\|_2^2 < 1$ (unit circle around $\mathbf{c}_1$)

(ii) $\|\mathbf{x} - \mathbf{c}_2\|_{S_2}^2 = (\mathbf{x} - \mathbf{c}_2)^\top \begin{pmatrix} 1.1 & -0.9 \\ -0.9 & 1.1 \end{pmatrix}^{-1} (\mathbf{x} - \mathbf{c}_2) < 1$ (ellipse around $\mathbf{c}_2$)
In order to find the smallest enclosing circle of the ellipse we can use the eigendecomposition of S_2 :
$\det(S_2 - \lambda \mathbf{I}) = 0 \iff \det \begin{pmatrix} 1.1-\lambda & -0.9 \\ -0.9 & 1.1-\lambda \end{pmatrix} = 0 \iff \lambda^2 - 2.2\lambda + 0.4 = 0 \iff \lambda_1 = 2.0, \lambda_2 = 0.2$
$\Rightarrow$ Eigenvalues μ_1, μ_2 of S_2^{-1} are $\mu_i = 1/\lambda_i$.
With this we get
$\|\mathbf{x} - \mathbf{c}_2\|_{S_2}^2 < 1 \iff (\mathbf{x} - \mathbf{c}_2)^\top \mathbf{V}^\top \begin{pmatrix} 5 & 0 \\ 0 & 0.5 \end{pmatrix} \mathbf{V}(\mathbf{x} - \mathbf{c}_2) < 1$ with $|\det \mathbf{V}| = 1$.
$\Rightarrow$ the circle around $\mathbf{c}_2$ with radius $\sqrt{1/0.5} = \sqrt{2}$ encloses the ellipse.
$\|\mathbf{c}_2 - \mathbf{c}_1\|_2 = \sqrt{2 \cdot 1.9^2} \approx 2.69 > 1 + \sqrt{2} \approx 2.41 \Rightarrow$ the circles can not intersect
$\Rightarrow$ the unit circle around $\mathbf{c}_1$ and the ellipse around $\mathbf{c}_2$ can not intersect $\Rightarrow$ only $\rho(\|\mathbf{x} - \mathbf{c}_1\|_{S_1})$ or $\rho(\|\mathbf{x} - \mathbf{c}_2\|_{S_2})$ can be non-zero for a given $\mathbf{x} \in \mathbb{R}^2$.

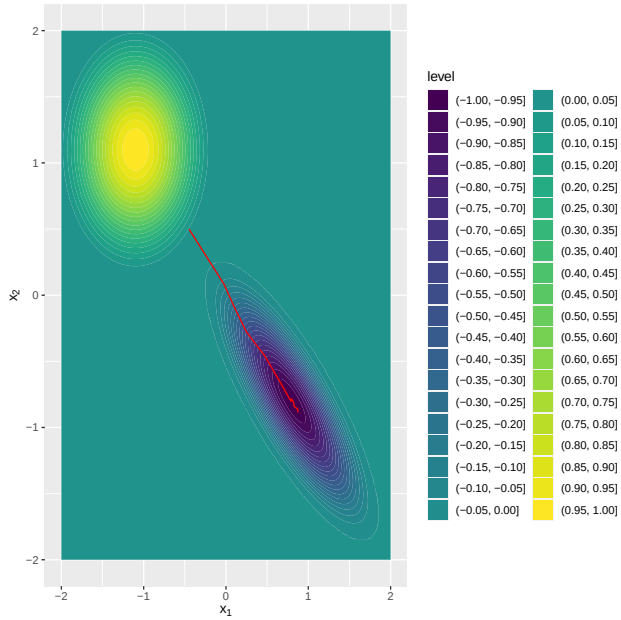
(c) Because of b) we know that we can treat $\rho(\|\mathbf{x}-\mathbf{c}_1\|_{S_1})$ and $\rho(\|\mathbf{x}-\mathbf{c}_2\|_{S_2})$ independently. Also it follows from $\rho(u) \geq 0 \; \forall u \in \mathbb{R}, w_1 > 0$ and $w_2 < 0$ that the global minimum must be in $\{\mathbf{x} \in \mathbb{R}^2 | \; \|\mathbf{x} - \mathbf{c}_2\|_{S_2}^2 < 1\}$
$\frac{\partial}{\partial \mathbf{x}} \rho(\|\mathbf{x}-\mathbf{c}_2\|_{S_2}) = 2(1-\|\mathbf{x}-\mathbf{c}_2\|_{S_2}^2) \cdot (-2) \cdot (\mathbf{x}-\mathbf{c}_2)^\top S_2^{-1} \overset{!}{=} \mathbf{0} \Rightarrow$ either $\|\mathbf{x}-\mathbf{c}_2\|_{S_2}^2 = 1$ (which is the boundary) or $\mathbf{x} = \mathbf{c}_2$.
Since $-\rho(1) = 0$ and $-\rho(\|\mathbf{c}_2 - \mathbf{c}_2\|) = -1 < 0$ it follows that the global minimum must be $\mathbf{x} = \mathbf{c}_2$.

(d)
```r
# we can treat the bump functions independently b)
grad <- function(x) {
  if((x - c1) %*% (x - c1) < 1){
    return(c(-4 * c(1 - (x - c1) %*% (x - c1)) * (x - c1)))
  }else if((x - c2) %*% S2_inv %*% (x - c2) < 1){
    return(c(4 * c(1 - (x - c2) %*% S2_inv %*%(x - c2)) * (x - c2) %*% S2_inv))
  }else{
    return(c(0, 0))
  }
}

alpha = 0.15

x0 = c(-0.45, 0.5)
x1 = x0 - alpha * grad(x0)
x2 = x1 - alpha * grad(x1)

print(x1)

## [1] -0.365175  0.421700

print(x2)

## [1] -0.365175  0.421700

print(grad(x1))

## [1] 0 0
```

We can not make any further progress with GD since the gradient is exactly zero.

(e) Start with $\mathbf{x}^{[0]} = (-0.45, 5)^\top$.
Since $\|\mathbf{c}_1 - \mathbf{x}^{[0]}\|_2^2 = 0.5525 < 1$ we know that $\nabla f(\mathbf{x}^{[0]}) = -4(1 - \|\mathbf{x} - \mathbf{c}_1\|_2^2) \cdot (\mathbf{x} - \mathbf{c}_1)^\top = (-0.5655, 0.5220)$.
$\mathbf{x}^{[1]} = \mathbf{x}^{[0]} - 0.15 * (-0.5655, 0.5220)^\top = (-0.3652, 0.422)^\top$.
Since $\|\mathbf{c}_1 - \mathbf{x}^{[1]}\|_2^2 = 1.0001 > 1$ and $\|\mathbf{c}_2 - \mathbf{x}^{[1]}\|_{S_2}^2 = 1.4323 > 1$ the gradient of f is zero at $\mathbf{x}^{[1]}$.
$\Rightarrow \mathbf{x}[2] = \mathbf{x}[1]$

(f)
```
alpha = 0.15

v = c(0.4, -0.4)
phi = 0.5
x = c(-0.45, 0.5)

xs = x
for (i in 1:15){
  v = phi * v - alpha*grad(x)
  x = x + v
  xs = rbind(xs, x)
}

cont_plot +
  geom_line(data = as.data.frame(xs), aes(x=V1, y=V2), color="red")
```



Multivariate Optimization 3

Exercise 1: Stochastic Gradient Descent

Consider the ordinary linear least squares problem (without intercept) where we want to minimize

$$\mathbb{E}_{\mathbf{x},y}[(\boldsymbol{\theta}^\top \mathbf{x} - y)^2]$$

with $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\mathbf{x})$ and $y|\mathbf{x} \sim \mathcal{N}({\boldsymbol{\theta}^*}^\top \mathbf{x}, \sigma^2)$.

(a) Show that $\mathbb{E}_{\mathbf{x},y}[\nabla_{\boldsymbol{\theta}}[(\boldsymbol{\theta}^\top \mathbf{x} - y)^2]] = \nabla_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x},y}[(\boldsymbol{\theta}^\top \mathbf{x} - y)^2]$

(b) Interpret a) in terms of SGD.

(c) Consider the univariate setting with $\boldsymbol{\Sigma}_\mathbf{x} = 1/4, \sigma = 1/10, \boldsymbol{\theta}^* = 1/2$ and a data set of size $10,000$.
Write an `R` script which plots the "confusion", i.e., the variance of the gradients, for $\theta \in \{0, 0.05, 0.1, \ldots 0.95, 1.0\}$. For each θ, plot 200 gradient samples.
Perform two such simulation studies with random batches of size 100 and $1,000$.

(d) What do you observe in c) ?

(e) Write an `R` script which solves the setting in c) with SGD with random batch sizes of 1 and $\alpha = 0.3$. Start with $\boldsymbol{\theta} = 0$ and perform 20 iterations. Repeat this process 200 times. Compare with GD.

Multivariate Optimization 3

Solution 1: Stochastic Gradient Descent

(a) We compute both expressions and compare the results.

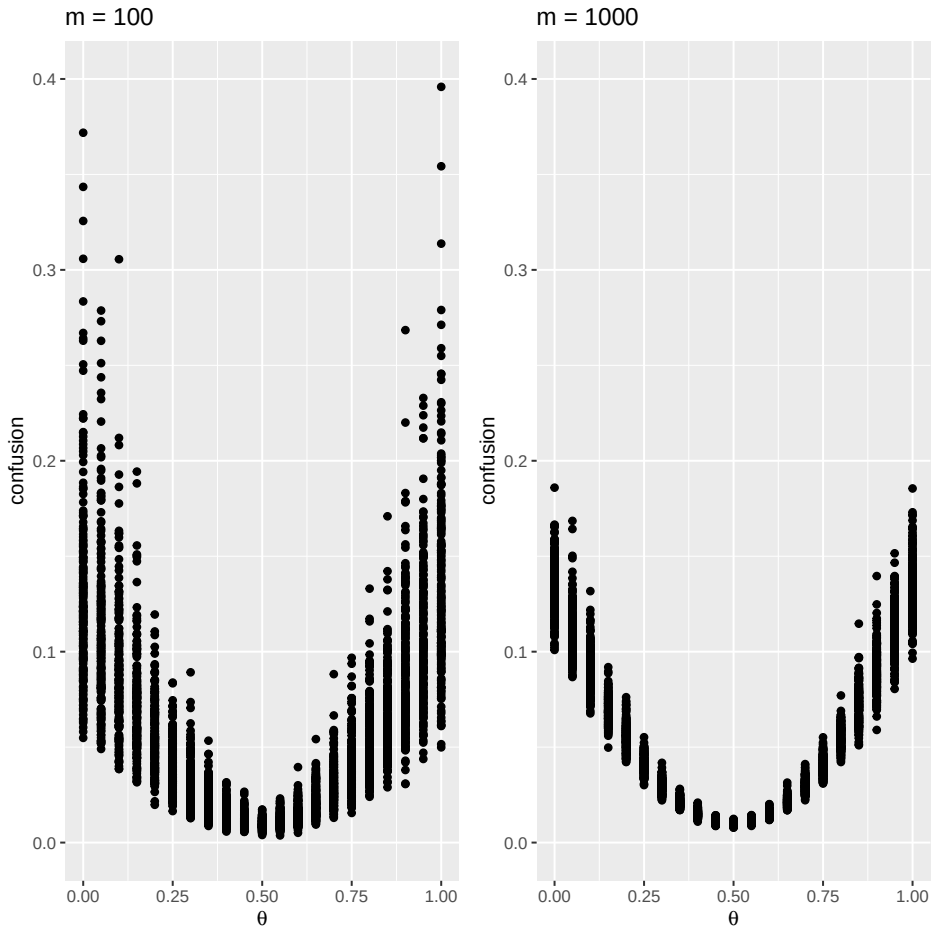
$$\begin{aligned}
\mathbb{E}_{\mathbf{x},y}[\nabla_{\boldsymbol{\theta}}[(\boldsymbol{\theta}^\top \mathbf{x} - y)^2]] &= \mathbb{E}_{\mathbf{x},y}[2\mathbf{x}\mathbf{x}^\top \boldsymbol{\theta} - 2\mathbf{x}y] \\
&= \mathbb{E}_{\mathbf{x}}\mathbb{E}_{y|\mathbf{x}}[2\mathbf{x}\mathbf{x}^\top \boldsymbol{\theta} - 2\mathbf{x}y] \\
&= \mathbb{E}_{\mathbf{x}}[2\mathbf{x}\mathbf{x}^\top \boldsymbol{\theta} - 2\mathbf{x}\mathbf{x}^\top \boldsymbol{\theta}^*] \\
&= 2\boldsymbol{\Sigma}_{\mathbf{x}}(\boldsymbol{\theta} - \boldsymbol{\theta}^*)
\end{aligned}$$

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}}\mathbb{E}_{\mathbf{x},y}[(\boldsymbol{\theta}^\top \mathbf{x} - y)^2] &= \nabla_{\boldsymbol{\theta}}\mathbb{E}_{\mathbf{x},y}[\boldsymbol{\theta}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{\theta} - 2\boldsymbol{\theta}^\top \mathbf{x}y + y^2] \\
&= \nabla_{\boldsymbol{\theta}}\mathbb{E}_{\mathbf{x}}[\boldsymbol{\theta}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{\theta}] - \nabla_{\boldsymbol{\theta}}\mathbb{E}_{\mathbf{x},y}[2\boldsymbol{\theta}^\top \mathbf{x}y] + \nabla_{\boldsymbol{\theta}}\mathbb{E}_y[y^2] \\
&= 2\Sigma_{\mathbf{x}}\boldsymbol{\theta} - 2\nabla_{\boldsymbol{\theta}}\mathbb{E}_{\mathbf{x}}\mathbb{E}_{y|\mathbf{x}}[\boldsymbol{\theta}^\top \mathbf{x}y] \\
&= 2\Sigma_{\mathbf{x}}\boldsymbol{\theta} - 2\nabla_{\boldsymbol{\theta}}\mathbb{E}_{\mathbf{x}}\mathbb{E}_{y|\mathbf{x}}[\boldsymbol{\theta}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{\theta}^*] \\
&= 2\Sigma_{\mathbf{x}}\boldsymbol{\theta} - 2\nabla_{\boldsymbol{\theta}}(\boldsymbol{\theta}^\top \Sigma_{\mathbf{x}}\boldsymbol{\theta}^*) \\
&= 2\Sigma_{\mathbf{x}}\boldsymbol{\theta} - 2\Sigma_{\mathbf{x}}\boldsymbol{\theta}^* \\
&= 2\boldsymbol{\Sigma}_{\mathbf{x}}(\boldsymbol{\theta} - \boldsymbol{\theta}^*)
\end{aligned}$$

(b) We can estimate $\mathbb{E}_{\mathbf{x},y}[\nabla_{\boldsymbol{\theta}}[(\boldsymbol{\theta}^\top \mathbf{x} - y)^2]]$ without bias via SGD, since we have access to realizations of gradients $\nabla_{\boldsymbol{\theta}}[(\boldsymbol{\theta}^\top \mathbf{x} - y)^2]$. From a), it follows that this estimate is also an unbiased estimate of the gradient of our objective function $\nabla_{\boldsymbol{\theta}}\mathbb{E}_{\mathbf{x},y}[(\boldsymbol{\theta}^\top \mathbf{x} - y)^2]$. Hence, SGD can be successfully applied in this situation.

(c)
```
library(ggplot2)
library(gridExtra)

set.seed(123)

sigma_x = 0.5
sigma_y = 0.1

n = 10000
x = sort(rnorm(n, sd = sigma_x))
theta_star = 0.5
y = theta_star * x + rnorm(n, sd = sigma_y)

theta = 0.9
mean(2*(x*x*theta - y*x))

## [1] 0.2015163

compute_conf <- function(theta, n){
  x = rnorm(n, sd = sigma_x)
  y = theta_star * x + rnorm(n, sd = sigma_y)
  # mean of squared differences between the sampled gradients and
  # the gradient of the objective
```

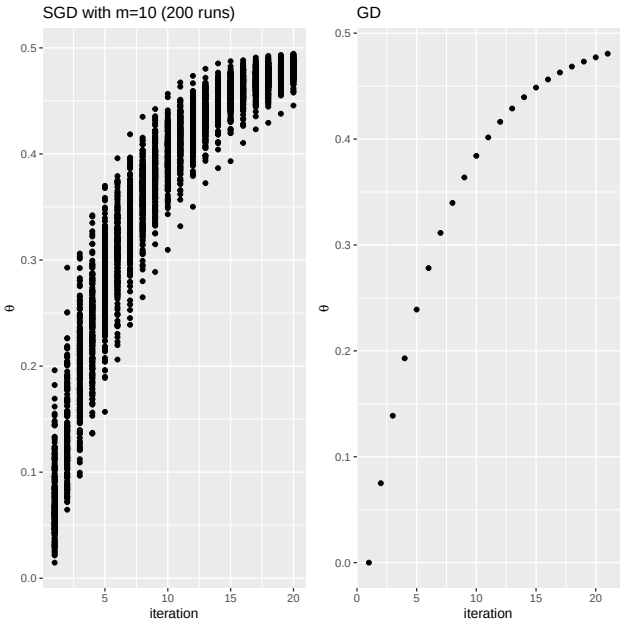
```r
  return(mean((2*(x*x*theta - y*x) - 2*sigma_x^2*(theta - theta_star))^2))
}

# compute confusions for m = 100

confs = c()
m = 100
reps = 200
thetas = seq(from=0, to=1, length.out = 21)
for(i in 1:reps){
  for(theta in thetas){
    confs = c(confs, compute_conf(theta, m))
  }
}

p_batch100 = ggplot(data.frame(thetas = rep(thetas, reps), confs = confs),
                    aes(x = thetas, y = confs)) +
  geom_point() + xlab(expression(theta)) + ylim(0, 0.4) + ggtitle("m = 100") +
  ylab("confusion")

# compute confusions for m = 1000

confs = c()
m = 1000
reps = 200
thetas = seq(from=0, to=1, length.out = 21)
for(i in 1:reps){
  for(theta in thetas){
    confs = c(confs, compute_conf(theta, m))
  }
}

p_batch1000 = ggplot(data.frame(thetas = rep(thetas, reps), confs = confs),
                     aes(x = thetas, y = confs)) +
  geom_point() + xlab(expression(theta)) + ylim(0, 0.4) + ggtitle("m = 1000")  +
  ylab("confusion")

# plot all
grid.arrange(p_batch100, p_batch1000, ncol = 2)
```



(d) Qualitatively, we observe for both settings that the mean and the variance of the confusion rise symmetrically around $\boldsymbol{\theta}^*$. As expected, the mean and the variance of the confusion is smaller for the larger batch size $m = 1000$ than for $m = 100$.

(e)
```
set.seed(123)

# SGD
thetas = NULL
alpha = 0.3
m = 10
for(j in 1:200){
  theta = 0
  for(i in 1:20){
    x = rnorm(m, sd = sigma_x)
    y = theta_star * x + rnorm(n, sd = sigma_y)

    theta = theta  - alpha * mean(2*(x*x*theta - y*x))
    thetas = rbind(thetas, theta)
  }
}

plot_sgd = ggplot(data.frame(thetas = thetas, it = rep(1:20, 200)),
       aes(x = it, y = thetas)) +
  geom_point() + ylab(expression(theta)) + xlab("iteration") +
  ggtitle("SGD with m=10 (200 runs)")

# GD
theta = 0
```

```
thetas = theta
alpha = 0.3

for(i in 1:20){
  theta = theta  - alpha *  2*sigma_x^2*(theta - theta_star)
  thetas = rbind(thetas, theta)
}

plot_gd = ggplot(data.frame(thetas = thetas, it = 1:21),
                 aes(x = it, y = thetas)) +
  geom_point() + ylab(expression(theta)) + xlab("iteration") + ggtitle("GD")

# plot all
grid.arrange(plot_sgd, plot_gd, ncol=2)
```

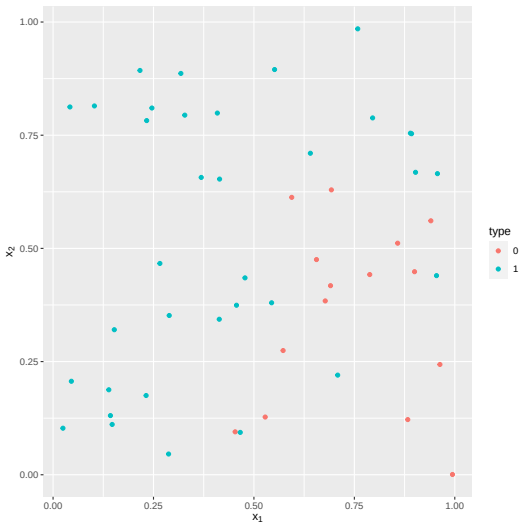


Multivariate Optimization 4

Exercise 1: Newton-Raphson and Gauss-Newton

You are given the following data situation:

```
library(ggplot2)

set.seed(123)

# simulate 50 binary observations with noisy linear decision boundary
n = 50
X = matrix(runif(2*n), ncol = 2)
X_model = cbind(1, X)
y = -((X_model %*% c(0.3, -1, 1) + rnorm(n, 0, 0.3) < 0) - 1)
df = as.data.frame(X)
df$type = as.character(y)

ggplot(df) +
  geom_point(aes(x = V1, y = V2, color=type)) +
  xlab(expression(x[1])) +
  ylab(expression(x[2]))
```



In the following we want to estimate a model $\pi : \mathbb{R}^2 \to [0,1], (x_1, x_2) \mapsto \frac{1}{1+\exp((1,x_1,x_2)^\top \boldsymbol{\theta})}$ such that it minimizes the Brier-loss, i.e., $\mathcal{R}_{\text{emp}} = \sum_{i=1}^{n} \|y^{(i)} - \pi(\mathbf{x}^{(i)})\|_2^2$.

(a) Show that the gradient

$$\nabla_{\boldsymbol{\theta}} \mathcal{R}_{\text{emp}} = \sum_{i=1}^{n} 2 \frac{y^{(i)}(\exp(f^{(i)})) - (\exp(-f^{(i)}) + 1)^{-1}}{(\exp(f^{(i)}) + 1)^2} \tilde{\mathbf{x}}^{(i)}$$

where $\tilde{\mathbf{x}}^{(i)} = (1, x_1^{(i)}, x_2^{(i)})^\top$ and $f^{(i)} = \tilde{\mathbf{x}}^{(i)\top} \boldsymbol{\theta}$

(b) Show that the Hessian $\nabla^2_{\boldsymbol{\theta}} \mathcal{R}_{\text{emp}} = \sum_{i=1}^{n} 2 \frac{\exp(f^{(i)})(y^{(i)}(-\exp(2f^{(i)})+1)-1+2\exp(f^{(i)}))}{(\exp(f^{(i)})+1)^4} \tilde{\mathbf{x}}^{(i)} \tilde{\mathbf{x}}^{(i)\top}$

(c) Show that $\mathcal{R}_{\text{emp}}$ is not convex in general

(d) Write an `R` script to find an optimal model via Newton-Raphson (do 30 iterations, $\mathbf{x}^{[0]} = \mathbf{0}$).

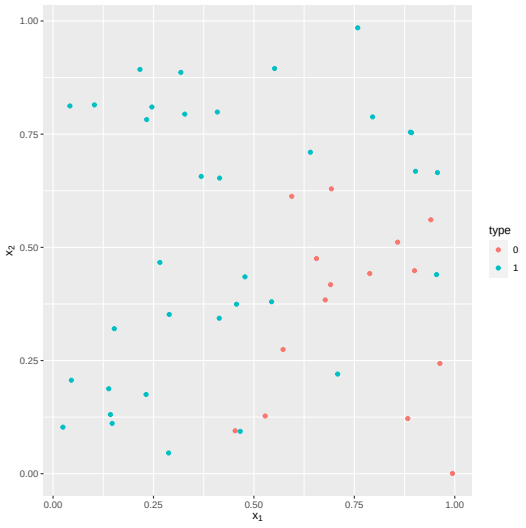
(e) Explain why Gauss-Newton is applicable here and write an `R` script to find an optimal model via Gauss-Newton (do 30 iterations, $\mathbf{x}^{[0]} = \mathbf{0}$).

Multivariate Optimization 4

Solution 1: Newton-Raphson and Gauss-Newton

You are given the following data situation:

```
library(ggplot2)

set.seed(123)

# simulate 50 binary observations with noisy linear decision boundary
n = 50
X = matrix(runif(2*n), ncol = 2)
X_model = cbind(1, X)
y = -((X_model %*% c(0.3, -1, 1) + rnorm(n, 0, 0.3) < 0) - 1)
df = as.data.frame(X)
df$type = as.character(y)

ggplot(df) +
  geom_point(aes(x = V1, y = V2, color=type)) +
  xlab(expression(x[1])) +
  ylab(expression(x[2]))
```



(a) Define $s : \mathbb{R} \to \mathbb{R}, f \mapsto \frac{1}{1+\exp(f)}$.

$\nabla_{\boldsymbol{\theta}} \mathcal{R}_{\text{emp}} = \nabla_{\boldsymbol{\theta}} \sum_{i=1}^{n} \|y^{(i)} - f(\mathbf{x}^{(i)})\|_2^2 = \sum_{i=1}^{n} \frac{d}{df} \|y^{(i)} - s(f^{(i)})\|_2^2 \cdot \nabla_{\boldsymbol{\theta}} f^{(i)}$

$= \sum_{i=1}^{n} 2 \frac{y^{(i)}(\exp(f^{(i)})+1)-1}{\exp(f^{(i)})+1} \cdot \frac{\exp(f^{(i)})}{(\exp(f^{(i)})+1)^2} \tilde{\mathbf{x}}^{(i)}$

$= \sum_{i=1}^{n} 2 \frac{y^{(i)}(\exp(f^{(i)})+1)-1}{\exp(f^{(i)})+1} \cdot \frac{\frac{\exp(f^{(i)})}{\exp(f^{(i)})+1}}{\exp(f^{(i)})+1} \tilde{\mathbf{x}}^{(i)}$

$= \sum_{i=1}^{n} 2 \frac{y^{(i)}(\exp(f^{(i)})) - \frac{\exp(f^{(i)})}{\exp(f^{(i)})+1}}{(\exp(f^{(i)})+1)^2} \tilde{\mathbf{x}}^{(i)}$

$= \sum_{i=1}^{n} 2 \frac{y^{(i)}(\exp(f^{(i)})) - (\exp(-f^{(i)})+1)^{-1}}{(\exp(f^{(i)})+1)^2} \tilde{\mathbf{x}}^{(i)}$

(b) $\nabla^2_{\boldsymbol{\theta}} \mathcal{R}_{\text{emp}} = \sum_{i=1}^{n} \frac{d}{df} 2 \frac{y^{(i)}(\exp(f^{(i)})) - (\exp(-f^{(i)})+1)^{-1}}{(\exp(f^{(i)})+1)^2} \tilde{\mathbf{x}}^{(i)} \nabla_{\boldsymbol{\theta}} f^{(i)\top}$

$= \sum_{i=1}^{n} 2 \frac{(y^{(i)}(\exp(f^{(i)})) - (\exp(-f^{(i)})+1)^{-2} \exp(-f^{(i)}))(\exp(f^{(i)})+1)^2 - (y^{(i)}(\exp(f^{(i)})) - (\exp(-f^{(i)})+1)^{-1}) \cdot 2(\exp(f^{(i)})+1) \exp(f^{(i)})}{(\exp(f^{(i)})+1)^4} \tilde{\mathbf{x}}^{(i)} \tilde{\mathbf{x}}^{(i)\top}$

$$= \sum_{i=1}^{n} 2\frac{y^{(i)}\exp(f^{(i)})(\exp(f^{(i)})+1)^2-\exp(f^{(i)})-(2y^{(i)}\exp(f^{(i)})(\exp(f^{(i)})+1)+2\exp(f^{(i)}))\exp(f^{(i)})}{(\exp(f^{(i)})+1)^4}\tilde{\mathbf{x}}^{(i)}\tilde{\mathbf{x}}^{(i)\top}$$
$$= \sum_{i=1}^{n} 2\frac{\exp(f^{(i)})(y^{(i)}(\exp(f^{(i)})+1)^2-1-2y^{(i)}\exp(f^{(i)})(\exp(f^{(i)})+1)+2\exp(f^{(i)}))}{(\exp(f^{(i)})+1)^4}\tilde{\mathbf{x}}^{(i)}\tilde{\mathbf{x}}^{(i)\top}$$

$$= \sum_{i=1}^{n} 2\frac{\exp(f^{(i)})(y^{(i)}(\exp(2f^{(i)})+2\exp(f^{(i)})+1-2\exp(2f^{(i)})-2\exp(f^{(i)}))-1+2\exp(f^{(i)}))}{(\exp(f^{(i)})+1)^4}\tilde{\mathbf{x}}^{(i)}\tilde{\mathbf{x}}^{(i)\top}$$

$$= \sum_{i=1}^{n} 2\frac{\exp(f^{(i)})(y^{(i)}(-\exp(2f^{(i)})+1)-1+2\exp(f^{(i)}))}{(\exp(f^{(i)})+1)^4}\tilde{\mathbf{x}}^{(i)}\tilde{\mathbf{x}}^{(i)\top}$$

(c) Assume, e.g., there is only one observation with $y^{(1)} = 0$ then

$$\nabla^2_{\boldsymbol{\theta}}\mathcal{R}_{\text{emp}} = \frac{2\exp(f^{(1)})(2\exp(f^{(1)})-1)}{(\exp(f^{(1)})+1)^4}\underbrace{\tilde{\mathbf{x}}^{(1)}\tilde{\mathbf{x}}^{(1)\top}}_{\text{p.s.d.}}.$$

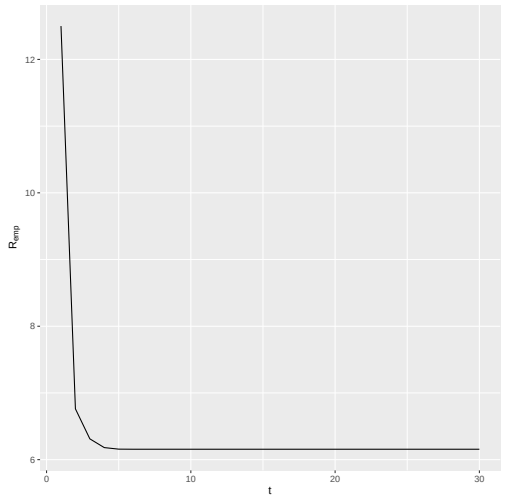
If a p.s.d. matrix is multiplied with a negative number it becomes a n.s.d. matrix, i.e., $\nabla^2_{\boldsymbol{\theta}}\mathcal{R}_{\text{emp}}$ is n.s.d. if $2\exp(f^{(1)}) < 1 \iff f^{(i)} < \ln(0.5)$. This condition trivially holds, e.g., if $\boldsymbol{\theta} = (\ln(0.5) - 1, 0, 0)^\top$.

(d) For Newton-Raphson, we need to solve in each update step

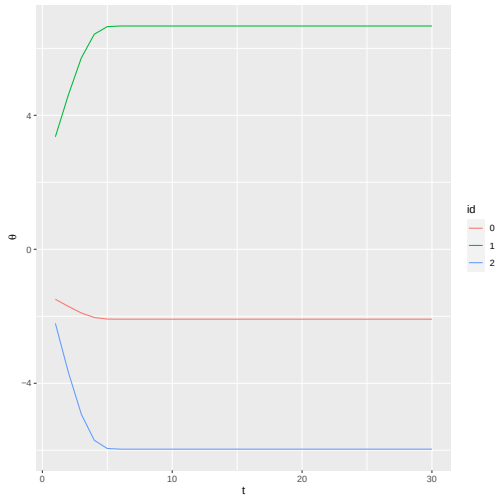
$$\nabla^2_{\boldsymbol{\theta}}\mathcal{R}_{\text{emp}}\mathbf{d} = -\nabla_{\boldsymbol{\theta}}\mathcal{R}_{\text{emp}}$$

for the descend direction $\mathbf{d}$.

```
theta = c(0, 0, 0)
remps = NULL
thetas = NULL

for(i in 1:30){
  exp_f = exp(X_model %*% theta)
  remps = rbind(remps, sum((y - 1/(1+exp_f))^2))

  hess = t(X_model) %*%
    (c((2 * exp_f*(2 * exp_f - y*(exp_f^2 - 1) - 1))/(exp_f + 1)^4 ) * X_model)
  grad = c(t(2*(y * exp_f - (1 + exp_f^-1)^-1) / (exp_f + 1)^2) %*% X_model)
  theta = theta + solve(hess, -grad)

  thetas = rbind(thetas, theta)
}

ggplot(data.frame(remps, t=1:nrow(remps)), aes(x=t, y=remps)) +
  geom_line() + ylab(expression(R[emp]))
```



```
ggplot(data.frame(theta = c(thetas), t=rep(1:nrow(thetas),3),
                  id = as.factor(rep(c(0, 1, 2), each= nrow(thetas)))),
       aes(x = t, y=theta)) +
  geom_line(aes(color = id)) + ylab(expression(theta))
```




```
theta

## [1] -2.087122  6.667438 -5.967500
```

(e) In this case, we can apply Gauss-Newton since $\mathcal{R}_{\text{emp}}$ is the squared sum of the residuals

$$\mathbf{r} = (y^{(1)} - \pi(\mathbf{x}^{(1)}), \ldots, y^{(n)} - \pi(\mathbf{x}^{(n)}))^\top.$$

Here, for the update step we need to compute

$$\nabla_{\boldsymbol{\theta}} \mathbf{r} = \begin{pmatrix} \frac{\exp(f^{(1)})}{(1+\exp(f^{(1)}))^2} \tilde{\mathbf{x}}^{(1)\top} \\ \vdots \\ \frac{\exp(f^{(n)})}{(1+\exp(f^{(n)}))^2} \tilde{\mathbf{x}}^{(n)\top} \end{pmatrix}$$

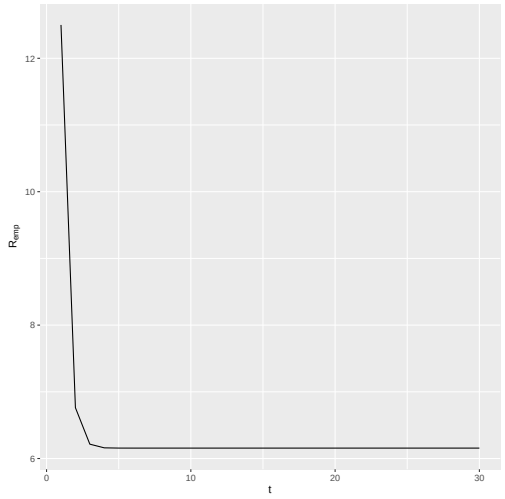
For Gauss-Newton, we solve in each update step

$$(\nabla_{\boldsymbol{\theta}} \mathbf{r}^\top \nabla_{\boldsymbol{\theta}} \mathbf{r}) \cdot \mathbf{d} = -\nabla_{\boldsymbol{\theta}} \mathbf{r}^\top \cdot \mathbf{r}$$

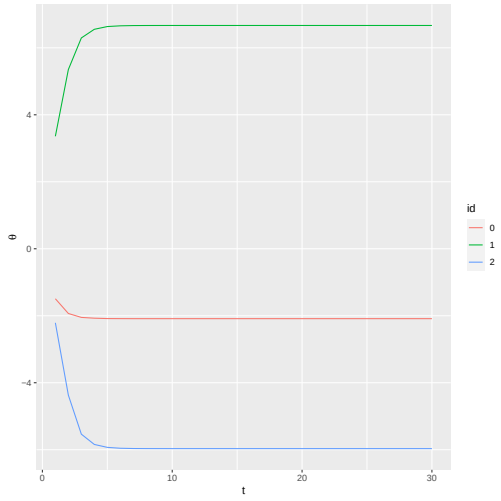
for the descend direction $\mathbf{d}$.

```
theta = c(0, 0, 0)
remps = NULL
thetas = NULL

for(i in 1:30){
  exp_f = exp(X_model %*% theta)
  remps = rbind(remps, sum((y - 1/(1+exp_f))^2))

  res = (y-1/(1+exp_f))
  grad_res = c(exp_f / (exp_f + 1)^2) * X_model

  theta = c(theta + solve(t(grad_res) %*% grad_res, -t(grad_res) %*% res))
  thetas = rbind(thetas, theta)
}
```

```
ggplot(data.frame(remps, t=1:nrow(remps)), aes(x=t, y=remps)) +
  geom_line() + ylab(expression(R[emp]))
```




```
ggplot(data.frame(theta = c(thetas), t=rep(1:nrow(thetas),3),
                  id = as.factor(rep(c(0, 1, 2), each= nrow(thetas)))),
       aes(x = t, y=theta)) +
  geom_line(aes(color = id)) + ylab(expression(theta))
```




```
theta
```

```
## [1] -2.087122  6.667438 -5.967500
```

Derivative Free Optimization and Evolutionary Strategies

Exercise 1: Coordinate Descent I
Minimize Ridge regression, i.e.,

$$\min_{\boldsymbol{\theta}} \frac{1}{2}\|\mathbf{X}\boldsymbol{\theta} - \mathbf{y}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2$$

for $\lambda \geq 0$ via coordinate descent under the assumption that $\mathbf{X}^\top\mathbf{X} = \mathbf{I}_d$.

Exercise 2: Coordinate Descent II
Consider the function

$$g : \mathbb{R}^2 \to \mathbb{R}, (x_1, x_2) \mapsto |x_1 - x_2| + 0.1(x_1 + x_2).$$

(a) Perform one round of coordinate descent starting from an arbitrary point (x_1, x_2). Show that after updating x_1 (while fixing x_2) and then updating x_2 (while fixing x_1) the algorithm arrives at a point where $x_1 = x_2$ and terminates. That is, show that coordinate descent will not move beyond the first iteration.

(b) Show that the global infimum of g is $-\infty$. Conclude that coordinate descent fails to find the true minimizer for this function.

Exercise 3: CMA-ES

Assume we have drawn the current population $\mathbf{x}_{1:\lambda}$ from the bivariate Gaussian distribution $\mathcal{N}(\mathbf{m}^{[0]}, \mathbf{C}^{[0]})$ with $\mathbf{m}^{[0]} = (1, 1)^\top, \mathbf{C}^{[0]} = \mathbf{I}$, such that

Id	x_1	x_2	Fitness value
1	1.14	0.24	0.67
2	1.54	-0.86	0.41
3	2.1	2.16	0.09
4	1.5	2.69	0.09
5	1.25	0.51	0.47
6	0.92	2.19	0.15

We want to do a simplified CMA-ES update step:

- Assume the parent number $\mu = 3$.
- Find $\mathbf{m}^{[1]}$ by updating $\mathbf{m}^{[0]}$ in the mean weighted[1] direction of $\mathbf{x}_{1:\mu}$ with stepsize 0.5.
- Compute $\mathbf{C}_\mu$, the (unweighted) sample covariance of $\mathbf{x}_{1:\mu}$ w.r.t. $\mathbf{m}^{[0]}$, and compute

  $$\mathbf{C}^{[1]} = (1 - c) \cdot \mathbf{C}^{[0]} + c \cdot \mathbf{C}_\mu$$

  with $c = 0.1$.

[1] Simply scale the fitness values such that they sum up to one.

Derivative Free Optimization and Evolutionary Strategies

Solution 1: Coordinate Descent I

$$\begin{aligned}
\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) &= \frac{1}{2}\|\mathbf{X}\boldsymbol{\theta} - \mathbf{y}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2 = \frac{1}{2}\mathbf{y}^\top\mathbf{y} - \mathbf{y}^\top\mathbf{X}\boldsymbol{\theta} + \frac{1}{2}\boldsymbol{\theta}^\top\boldsymbol{\theta} + \frac{\lambda}{2}\boldsymbol{\theta}^\top\boldsymbol{\theta} \\
&= \frac{1}{2}\mathbf{y}^\top\mathbf{y} - \sum_{j=1}^{d}\mathbf{y}^\top\mathbf{x}_j\theta_j + \frac{1}{2}(1+\lambda)\boldsymbol{\theta}^\top\boldsymbol{\theta} \\
\frac{\partial\mathcal{R}_{\text{emp}}}{\theta_j} &= (1+\lambda)\theta_j - \mathbf{y}^\top\mathbf{x}_j \overset{!}{=} 0 \\
&\Rightarrow \theta_j^* = \frac{\mathbf{y}^\top\mathbf{x}_j}{1+\lambda}
\end{aligned}$$

Solution 2: Coordinate Descent II

(a) Update x_1 whle fixing x_2: We fix $x_2 = c$ (constant). The function then states as

$$g(x_1, c) = |x_1 - c| + 0.1(x_1 + c).$$

We want to choose x_1 to minimize this. Due to the absolute value, there are two cases.

(i) Case 1: $x_1 \geq c$: Then $|x_1 - c| = x_1 - c$. So $g(x_1, c) = (x_1 - c) + 0.1x_1 + 0.1c = 1.1x_1 - 0.9c$. As a function of x_1 this is strictly increasing (derivative of $1.1 > 0$). Therefore, the minimizer given $x_1 \geq c$ is at the left boundary, i.e., $x_1 = c$.

(ii) Case 2: $x_1 < c$: Then $|x_1 - c| = c - x_1$. So $g(x_1, c) = (c - x_1) + 0.1x_1 + 0.1c = 1.1c - 0.9x_1$. As a function of x_1 this is strictly decreasing (derivative of $-0.9 < 0$). Thereore, the minimizer given $x_1 < c$ is at the right boundary, i.e., $x_1 = c$.

In both cases, the best choice is $x_1^* = c$. Note that x_2 was fixed to be c, i.e., the function is minimized exactly when $x_1 = x_2$.
After updating x_1 while holding x_2 constant, we arrive at $(x_1^{[1]}, x_2^{[0]}) = (x_2^{[0]}, x_2^{[0]})$.

Update x_2 while fixing x_1: We now fix $x_1 = c$ (constant). The function then states as

$$g(c, x_2) = |c - x_2| + 0.1(c + x_2).$$

Note that g is symmetric in its arguments, therefore based on the first analysis, we conclude that again $x_2^* = c$.
After updating x_2 while holding x_1 constant, we arrive at $(x_1^{[1]}, x_2^{[1]}) = (x_1^{[1]}, x_1^{[1]}) = (x_2^{[0]}, x_2^{[0]})$.

We observe that coordinate updates will set the respective coordinate to the value of the other constant held coordinate value and once the algorithm arrives at $x_1 = x_2 = c$, neither coordinate update will move the point.

(b) Along the diagonal $x_1 = x_2 = t$, the function simplifies to

$$g(t, t) = |t - t| + 0.1(t + t) = 0.2t.$$

As $t \to -\infty$, $0.2t \to -\infty$, hence the infimum of g is $-\infty$. No finite (x_1, x_2) can achieve that inifimum, i.e., there is no global minimizer, but the values of g can be made arbitrarly negative by letting x_1, x_2 be arbitrarly negative.

Solution 3: CMA-ES

Pick $\mu = 3$ parents with highest fitness values, i.e., Id $= 1, 2, 5$ which we denote with $\mathbf{x}_{1:\mu}$ and respective weights $w_i = \frac{f_i}{\sum_{i=1}^{\mu} f_i} \approx (0.432, 0.265, 0.303)$.

$$
\begin{aligned}
\mathbf{m}^{[1]} &= \mathbf{m}^{[0]} + 0.5 \sum_{i=1}^{3} w_i (\mathbf{x}_i - \mathbf{m}^{[0]}) \approx (1.140, 0.515)^\top \\
\mathbf{C}_\mu &= \frac{1}{3-1} \sum_{i=1}^{3} (\mathbf{x}_i - \mathbf{m}^{[0]}))(\mathbf{x}_i - \mathbf{m}^{[0]}))^\top \\
&\approx \begin{pmatrix} 0.187 & -0.617 \\ -0.617 & 2.139 \end{pmatrix} \\
\mathbf{C}^{[1]} &= 0.9 \cdot \mathbf{I}_d + 0.1 \cdot \mathbf{C}_\mu \\
&\approx \begin{pmatrix} 0.919 & -0.062 \\ -0.062 & 1.114 \end{pmatrix}
\end{aligned}
$$

Bayesian Optimization

Exercise 1: Expected Improvement

Derive the closed form expression of the Expected Improvement:

$$a_{\text{EI}}(\mathbf{x}) = \left(f_{\min} - \hat{f}(\mathbf{x})\right) \Phi\left(\frac{f_{\min} - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) + \hat{s}(\mathbf{x})\phi\left(\frac{f_{\min} - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right).$$

Assume that $Y(\mathbf{x}) \sim \mathcal{N}\left(\hat{f}(\mathbf{x}), \hat{s}^2(\mathbf{x})\right)$.

Hints:

- For notational clarity, let's introduce y for the random variable $Y(\mathbf{x})$ and $p(y) \coloneqq P(Y|\mathbf{x}, \mathcal{D}^{[t]}) = \mathcal{N}\left(\hat{f}(\mathbf{x}), \hat{s}^2(\mathbf{x})\right)$ for its probability density function.
- Start with $a_{\text{EI}}(\mathbf{x}) = \mathbb{E}_y(\max\{f_{\min} - y, 0\}) = \int_{-\infty}^{\infty} \max\{f_{\min} - y, 0\} p(y) dy$.
- Decompose the integral additively depending on whether $y < f_{\min}$ or $y \geq f_{\min}$ to get rid of the maximum operator.
- It is helpful to substitute y by $u \coloneqq \frac{y - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}$ which implies that $y = u\hat{s}(\mathbf{x}) + \hat{f}(\mathbf{x})$. This allows you to work with standard normal distributions. Note however, that this implies performing a change of variable within the integral.
- Denote the standard normal probability density function by $\phi(z) = \frac{1}{\sqrt{2\pi}} \exp\left(\frac{-z^2}{2}\right)$ and the standard normal cumulative distribution function by $\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} \exp\left(\frac{-u^2}{2}\right) du$.
- There is a useful identity: $\int_{-\infty}^{z} u\phi(u) du = -\phi(z)$.

Exercise 2: BO Loop

We want to implement our own BO algorithm using a Gaussian Process (GP) as surrogate model and Expected Improvement as acquisition function. Our goal is to minimize the following univariate function:

$$f : [0, 1] \to \mathbb{R}, x \mapsto 2x \cdot \sin(14x).$$

We start with an initial design of 4 points sampled uniformly at random.

(a) Write down the BO algorithm in pseudocode style.

(b) Implement the algorithm. For the GP you can for example use the `DiceKriging` package (see `?DiceKriging::km`). Use an RBF kernel. Optimize the Expected Improvement via a univariate method such as Brent's method (see `?optimize`). Use your BO algorithm to minimize f and terminate after 10 function evaluations in total.

Bayesian Optimization

Exercise 1: Expected Improvement

We start with

$$a_{\text{EI}}(\mathbf{x}) = \mathbb{E}_y(\max\{f_{\min} - y, 0\}) = \int_{-\infty}^{\infty} \max\{f_{\min} - y, 0\} p(y) dy.$$

Observe that

$$\max\{f_{\min} - y, 0\} = \begin{cases} f_{\min} - y, & \text{if } y < f_{\min}, \\ 0, & \text{otherwise.} \end{cases}$$

All contributions for $y \geq f_{\min}$ are zero. Therefore, we can additively decompose the integral and it simplifies to

$$a_{\text{EI}}(\mathbf{x}) = \int_{-\infty}^{f_{\min}} (f_{\min} - y)\, p(y) dy.$$

$$\begin{aligned}
\alpha_{\text{EI}}(\mathbf{x}) &= \int_{-\infty}^{f_{\min}} (f_{\min} - y)\, p(y) dy \\
&= \int_{-\infty}^{f_{\min}} (f_{\min} - y)\, \frac{1}{\sqrt{2\pi \hat{s}(\mathbf{x})^2}} \exp\left(-\frac{\left(y - \hat{f}(\mathbf{x})\right)^2}{2\hat{s}(\mathbf{x})^2} \right) dy \\
&= \int_{-\infty}^{z} \left(f_{\min} - \hat{f}(\mathbf{x}) - u\hat{s}(\mathbf{x})\right) \frac{1}{\sqrt{2\pi \hat{s}(\mathbf{x})^2}} \exp\left(-\frac{u^2}{2}\right) \hat{s}(\mathbf{x}) du \quad \left(\text{Def. } u := \frac{y - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}, \frac{du}{dy} = \frac{1}{\hat{s}(\mathbf{x})}, z := \frac{f_{\min} - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) \\
&= \int_{-\infty}^{z} \left(f_{\min} - \hat{f}(\mathbf{x}) - u\hat{s}(\mathbf{x})\right) \phi(u) du \\
&= \int_{-\infty}^{z} \left(f_{\min} - \hat{f}(\mathbf{x})\right) \phi(u) du - \int_{-\infty}^{z} (u\hat{s}(\mathbf{x}))\, \phi(u) du
\end{aligned}$$

Note that

$$\Phi(z) = \int_{-\infty}^{z} \phi(u) du$$

by definition.

Therefore, regarding the first integral:

$$\int_{-\infty}^{z} \left(f_{\min} - \hat{f}(\mathbf{x})\right) \phi(u) du = \left(f_{\min} - \hat{f}(\mathbf{x})\right) \Phi(z) = z\hat{s}(\mathbf{x})\Phi(z).$$

Regarding the second integral we use the identity

$$\int_{-\infty}^{z} u\phi(u) du = -\phi(z).$$

Putting both together we obtain:

$$\begin{aligned}
\alpha_{\text{EI}}(\mathbf{x}) &= z\hat{s}(\mathbf{x})\Phi(z) - \hat{s}(\mathbf{x})(-\phi(z)) \\
&= z\hat{s}(\mathbf{x})\Phi(z) + \hat{s}(\mathbf{x})\phi(z) \\
&= \left(f_{\min} - \hat{f}(\mathbf{x})\right) \Phi\left(\frac{f_{\min} - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) + \hat{s}(\mathbf{x})\phi\left(\frac{f_{\min} - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right).
\end{aligned}$$

Exercise 2: BO Loop

(a) Let $\mathcal{D}$ be the initial design consisting of $\{(x^{[1]}, y^{[1]}), \ldots, (x^{[4]}, y^{[4]})\}$. Set t to 4.
While $t < 10$:

(i) Fit surrogate model on $\mathcal{D}$.

(ii) Optimize the Expected Improvement $a_{\text{EI}}(x)$ to obtain a new point $x^{[t+1]} := \arg\max_{x \in [0,1]} a_{\text{EI}}(x)$.

(iii) Evaluate $x^{[t+1]}$ and update design data $\mathcal{D} = \mathcal{D} \cup \{(x^{[t+1]}, f(x^{[t+1]}))\}$.

(iv) Set t to $t + 1$.

Return x that minimizes $f(x)$ in $\mathcal{D}$: $\arg\min_{(x,y) \in \mathcal{D}} y$.

(b)
```r
library(DiceKriging)
set.seed(0308)
f = function(x) 2*x * sin(14*x)
initial_x = runif(4, min = 0, max = 1)
initial_y = f(initial_x)
design = data.frame(x = initial_x, y = initial_y)
t = 4

ei = function(x, current_fmin, current_gp) {
  gp_prediction = predict(current_gp, newdata = data.frame(x = x), type="SK")
  gp_mean = gp_prediction$mean
  gp_sd = gp_prediction$sd
  diff = (current_fmin - gp_mean)
  z = diff / gp_sd
  diff * pnorm(z) + gp_sd * dnorm(z)
}


while (t < 10) {
  gp = km(design = design[, 1L, drop = FALSE], response = design[, 2L],
        covtype = "gauss", nugget = 1e-8)
  fmin = min(design$y)
  x_new = optimize(f = ei, interval = c(0, 1), maximum = TRUE,
        current_fmin = fmin, current_gp = gp)$maximum
  design = rbind(design, data.frame(x = x_new, y = f(x_new)))
  t = t + 1
}

design[which.min(design$y), ]
```

Multi-Criteria Optimization

Exercise 1: Concepts in Multi-Criteria Optimization

Analyse a Multi-Objective Optimization problem with the following six points and values in objective space:

$$\mathbf{x}^{(1)} \text{ with } \mathbf{f}^{(1)} = (10, 5)$$
$$\mathbf{x}^{(2)} \text{ with } \mathbf{f}^{(2)} = (7, 8)$$
$$\mathbf{x}^{(3)} \text{ with } \mathbf{f}^{(3)} = (4, 6)$$
$$\mathbf{x}^{(4)} \text{ with } \mathbf{f}^{(4)} = (6, 4)$$
$$\mathbf{x}^{(5)} \text{ with } \mathbf{f}^{(5)} = (9, 3)$$
$$\mathbf{x}^{(6)} \text{ with } \mathbf{f}^{(6)} = (3, 7)$$

(a) Determine which of these points are Pareto optimal (find $\mathcal{P}$).

(b) Sketch the objective space and indicate the Pareto front $f(\mathcal{P})$.

(c) Assume a reference point $R = (15, 15)$. Calculate the dominated hypervolume of the Pareto optimal points.

(d) Perform non-dominated sorting.

(e) Compute the crowding distance of the point $\mathbf{x}^{(3)}$ with the solution $\mathbf{f}^{(3)}$.

(f) Compute the hypervolume contribution of the point $\mathbf{x}^{(5)}$ with the solution $\mathbf{f}^{(5)}$. Again, assume a reference point $R = (15, 15)$.

Note: We want to minimize both objectives f_1, f_2.

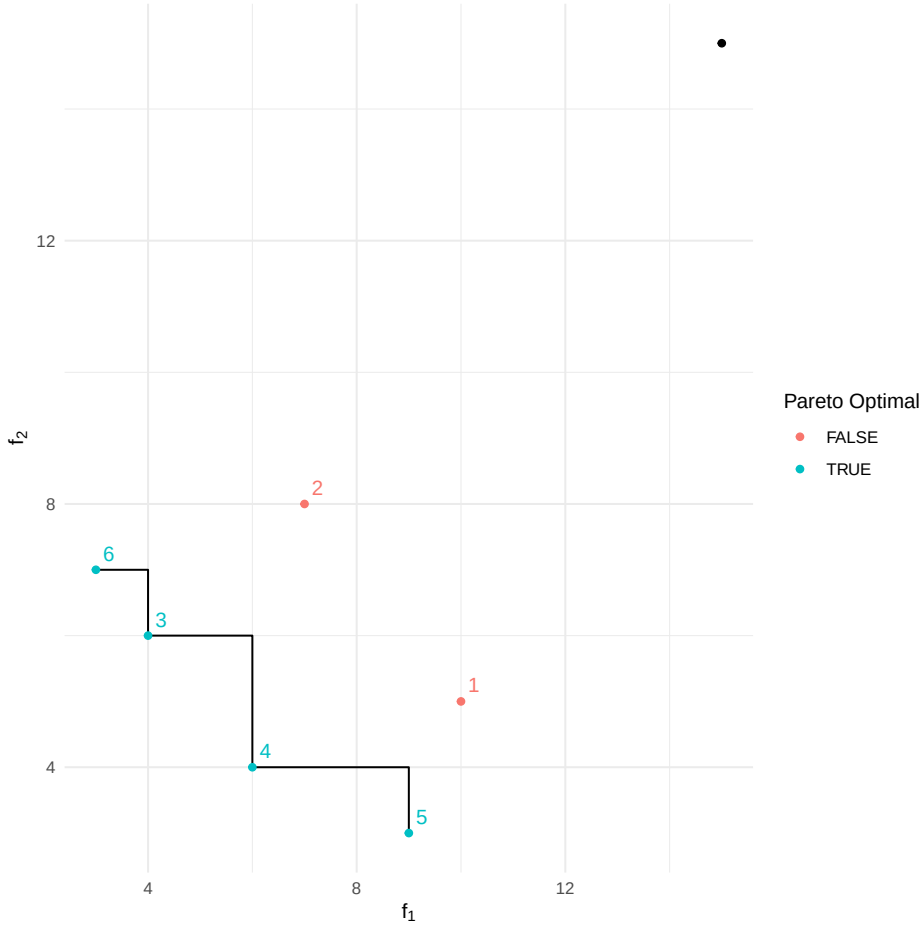
Multi-Criteria Optimization

Exercise 1: Concepts in Multi-Criteria Optimization

(a)
- $\mathbf{x}^{(1)}$ with $\mathbf{f}^{(1)} = (10, 5)$ e.g., dominated by $\mathbf{x}^{(4)}$ with $\mathbf{f}^{(4)} = (6, 4)$.
- $\mathbf{x}^{(2)}$ with $\mathbf{f}^{(2)} = (7, 8)$ e.g., dominated by $\mathbf{x}^{(3)}$ with $\mathbf{f}^{(3)} = (4, 6)$.
- $\mathbf{x}^{(3)}$ with $\mathbf{f}^{(3)}$ not dominated.
- $\mathbf{x}^{(4)}$ with $\mathbf{f}^{(4)}$ not dominated.
- $\mathbf{x}^{(5)}$ with $\mathbf{f}^{(5)}$ not dominated.
- $\mathbf{x}^{(6)}$ with $\mathbf{f}^{(6)}$ not dominated.

$\rightarrow$ the set of Pareto optimal points is $\mathcal{P} = \{\mathbf{x}^{(3)}, \mathbf{x}^{(4)}, \mathbf{x}^{(5)}, \mathbf{x}^{(6)}\}$.

(b)
```r
library(ggplot2)

solutions = data.frame(f1 = c(10, 7, 4, 6, 9, 3), f2 = c(5, 8, 6, 4, 3, 7), id = 1:6)
solutions$pareto_optimal = c(FALSE, FALSE, TRUE, TRUE, TRUE, TRUE)

ggplot(aes(x = f1, y = f2, colour = pareto_optimal), data = solutions) +
  geom_step(data = solutions[solutions$pareto_optimal == TRUE, ],
            direction = "hv", colour = "black") +
  geom_point() +
  geom_text(aes(x = f1, y = f2, label = id),
            nudge_x = 0.25, nudge_y = 0.25, show.legend = FALSE) +
  geom_point(aes(x = f1, y = f1), colour = "black", data = data.frame(f1 = 15, f2 = 15)) +
  labs(x = expression(f[1]), y = expression(f[2]), colour = "Pareto Optimal") +
  theme_minimal()
```



(c) We can simply compute the area slices under each segment and sum them up.
For the four rectangles from left to right:

- $(4-3)\cdot(15-7)=8$
- $(6-4)\cdot(15-6)=18$
- $(9-6)\cdot(15-4)=33$
- $(15-9)\cdot(15-3)=72$

$\rightarrow S(\mathcal{P},R)=8+18+33+72=131$.

(d) We start with the first front of non-dominated solutions $\mathcal{F}_1=\{\mathbf{x}^{(3)},\mathbf{x}^{(4)},\mathbf{x}^{(5)},\mathbf{x}^{(6)}\}$. After dropping these solutions, $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ remain. Neither of these solutions dominates the other solution. Therefore $\mathcal{F}_2=\{\mathbf{x}^{(1)},\mathbf{x}^{(2)}\}$.

(e) Crowding distance is always computed within a front. From (d) we have that $\mathbf{x}^{(3)}\in\mathcal{F}_1$. We start with the first dimension, f_1.
First, sort the values by f_1: $(3,7),(4,6),(6,4),(9,3)$. $(3,7)$ and $(9,3)$ are outermost and get an infinite partial distance for the f_1 dimension. Normalize the values by the minimum of 3 and maximum of 9 among the four points. For the point $\mathbf{x}^{(3)}$ (new index of $i=2$) we compute:

$$\mathrm{CD}_1(\mathbf{x}^{(3)})=\frac{(f_1^{(i+1)}-f_1^{(i-1)})}{(f_1^{(\max)}-f_1^{(\min)})}=\frac{(6-3)}{(9-3)}=0.5.$$

For the second dimension, f_2, we analogously obtain:

$$\mathrm{CD}_2(\mathbf{x}^{(3)})=\frac{(f_2^{(i+1)}-f_2^{(i-1)})}{(f_2^{(\max)}-f_2^{(\min)})}=\frac{(7-4)}{(7-3)}=0.75.$$

$\rightarrow$ the total crowding distance is (when taking the sum) $0.5+0.75=1.25$.

(f) We know from (c) that the total dominated hypervolume is $S(\mathcal{P}, R) = 131$. To compute the hypervolume contribution of $\mathbf{x}^{(5)}$, we compute the hypervolume of $\mathcal{P} \setminus \mathbf{x}^{(5)}$ and substract it. Similar computations as in (c) but now for $\mathcal{P} \setminus \mathbf{x}^{(5)}$ yield $S(\mathcal{P} \setminus \mathbf{x}^{(5)}, R) = 125$. Therefore $\mathbf{x}^{(5)}$ has a hypervolume contribution of $131 - 125 = 6$.