Advanced Statistical Computing
Week 6: Numerical Integration, Convex Optimization

Aad van der Vaart

Fall 2012

Contents

Numerical Integration

Problem

For a known function f we wish to find a good approximation to

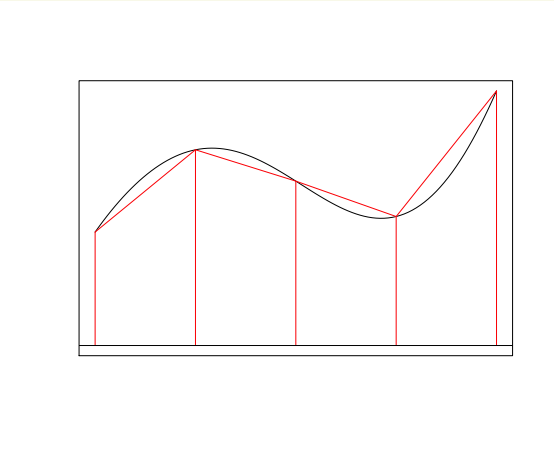
$$\int_a^b f(x)\,dx.$$

Stochastic optimization, possibly by *importance sampling*, is often easy to implement, but has the disadvantage that the error is random.

Numerical integration is deterministic, and often comes with an error estimate.

- Trapezium rule
- Simpson's rule
- Gaussian quadrature
- ... many refinements, variants, acceleration methods,....

Trapezium rule

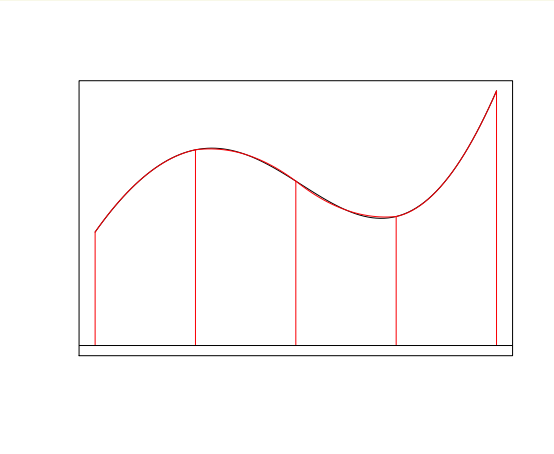


For $x_0 = a, x_1 = a + h, x_2 = a + 2h, \ldots, x_n = a + nh = b$:

$$\int_a^b f(x)\,dx \approx h\Big(\tfrac{1}{2}f(a) + f(x_1) + f(x_2)\cdots + f(x_{n-1}) + f(b)/2\Big).$$

```
> f=function(u) -sin(1+u^2)^2*u^2/(0.5+u^3)-0.2*u^2+u+0.2
> u=seq(0,1,by=0.25); n=length(u)
> 0.25*(f(u[1])/2+sum(f(u[-c(1,n)]))+f(u[n])/2)
[1] 0.2966339
> for (n in c(10,100, 1000,10000, 100000,1000000)){
+   u=seq(0,1,by=1/n)
+   print((1/n)*(f(u[1])/2+sum(f(u[-c(1,n)]))+f(u[n])/2))
+ }
[1] 0.3005307
[1] 0.2934621
[1] 0.2933787
[1] 0.2933779
[1] 0.2933778
[1] 0.2933778
```

Simpson's rule



For $x_0 = a, x_1 = a + h, x_2 = a + 2h, \ldots, x_{2n} = a + 2nh = b$,:

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\Big(f(a) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \cdots + 4f(x_{2n-1}) + f(b)\Big).$$

```
> f=function(u) -sin(1+u^2)^2*u^2/(0.5+u^3)-0.2*u^2+u+0.2
> u=seq(0,1,by=0.25); n=length(u)
> 0.25/6*(f(u[1])+f(u[n])+4*sum(f((u[-n]+u[-1])/2))+2*sum(f(u[-c(1,n)])))
[1] 0.2933571
> for (n in c(10,100, 1000,10000, 100000,1000000)){u=seq(0,1,by=1/n)
+  print(1/(n*6)*(f(u[1])+f(u[n])+4*sum(f((u[-n]+u[-1])/2))+2*sum(f(u[-c(1,n)]))))    }
[1] 0.3005023
[1] 0.2934574
[1] 0.2933786
[1] 0.2933779
[1] 0.2933778
[1] 0.2933778
```

Gauss quadrature

Gauss quadrature also gives an approximation of the form

$$\int_a^b f(x)\,dx \approx \sum_{i=1}^{n} v_i f(x_i).$$

However, the x_i are *not* a regular grid, but zeros of the nth *Hermite polynomial*.

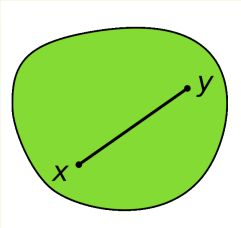
The weights v_i are chosen to make the approximation exact for all f such that $f(x)e^{x^2}$ is a polynomial of degree $\leq 2n-1$.

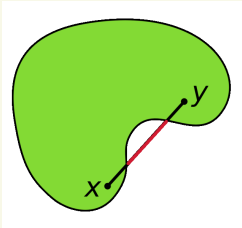
The `R` function `integrate` implements a variant, *Gauss-Kronrod*, for which the points x_i for different n are nested.

```
> f=function(u) -sin(1+u^2)^2*u^2/(0.5+u^3)-0.2*u^2+u+0.2
> integrate(f,0,1)
0.2933778 with absolute error < 2.5e-11
```

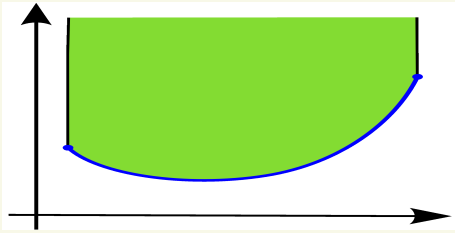
Convex Optimization

Convex domains and functions





A subset $\Omega \subset \mathbb{R}^n$ is *convex* if $\lambda x + (1-\lambda)y \in \Omega$ whenever $x, y \in \Omega$ and $0 \le \lambda \le 1$.



A function $f\colon \Omega \to \mathbb{R}$ is *convex* if the area above its graph is convex, or more technically

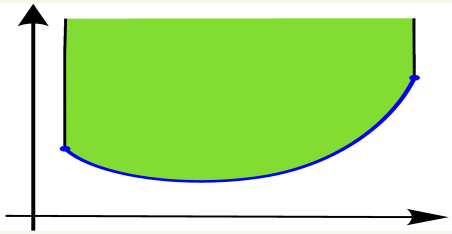
$$f\big(\lambda x + (1-\lambda)y\big) \le \lambda f(x) + (1-\lambda) f(y), \qquad \forall x, y \in \Omega.$$

A function is *strictly convex* if the inequality is strict.

EXAMPLES: $x \mapsto x$, $x \mapsto x^2$, $x \mapsto -\log x$, $(x,y) \mapsto x^2 + y^2$, $x \mapsto x^T Q x$ for $Q \ge 0$.

[A twice continuously differentiable function is convex iff the Hessian $D^2 f$ is nonnegative-definite.]

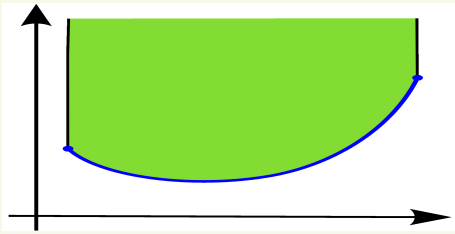
Optimizing a convex function: local is global



THEOREM
If f is convex with a *local* minimum at x_0, then f has a *global* minimum at x_0. Furthermore, the set of points of minimum is convex, and consists of at most one point if f is strictly convex.

CONSEQUENCE
Convex criterium functions are nice for *minimization*.

Optimizing a convex function: local is global



THEOREM
If f is convex with a *local* minimum at x_0, then f has a *global* minimum at x_0. Furthermore, the set of points of minimum is convex, and consists of at most one point if f is strictly convex.

CONSEQUENCE
Convex criterium functions are nice for *minimization*.

PROOF If $f(x) < f(x_0)$, then $f\big(\lambda x + (1-\lambda)x_0\big) \leq \lambda f(x) + (1-\lambda) f(x_0) < f(x_0)$, for any $0 \leq \lambda \leq 1$. But for $\lambda \approx 0$ the point $\lambda x + (1-\lambda)x_0$ is near x_0, contradicting that x_0 is a local minimum.

[More precise statement: if $f\colon \Omega \to \mathbb{R}$ is convex on the convex domain Ω and $f(x_0) \leq f(x)$ for every $x \in \Omega$ with $\|x - x_0\| < r$ for some $r > 0$, then $f(x_0) \leq f(x)$ for all $x \in \Omega$.]

Example — sparsity

If in linear regression $Y = X\beta + e$ it is known that many β_i are zero, then it is reasonable to estimate β *not* by OLS, but by

$$\hat{\beta} = \operatorname*{argmin}_{\beta} \Big[\|Y - X\beta\|^2 + \lambda \big(\#(i \colon \beta_i \neq 0) \big) \Big]. \tag{1}$$

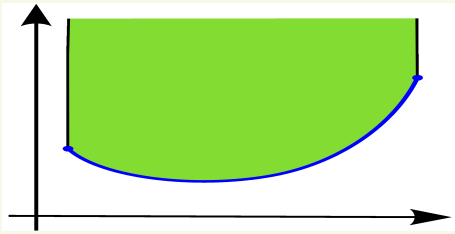
Unfortunately, this is not convex in β and difficult to compute in high dimensions. An alternative is

$$\operatorname*{argmin}_{\beta} \Big[\|Y - X\beta\|^2 + \lambda \sum_{i=1}^{p} |\beta_i|^\gamma \Big]. \tag{2}$$

For $\gamma \downarrow 0$ this approaches (1). For $\gamma \geq 1$ this is convex.

[For $\lambda = 2\sigma^2$ criterion (1) is a version of AIC. For $\gamma = 1$ criterion (2) is the LASSO; for $\gamma = 2$ ridge regression.]

Convex program



For given convex $f, g_1, \dots g_m \colon \Omega \to \mathbb{R}$

$$\min\Big\{ f(x) \colon \qquad x \in \Omega, \quad g_i(x) \le 0, \quad i = 1, 2, \dots, m \Big\}.$$

ALGORITHMS

- Linear
- Quadratic
- Many other, general purpose and for special subproblems

Kuhn-Tucker theorem

For given convex $f, g_1, \ldots, g_m \colon \Omega \to \mathbb{R}$,

$$V := \min\Big\{ f(x) \colon \qquad x \in \Omega, \quad g_i(x) \le 0, \;\; i = 1, 2, \ldots, m \Big\}.$$

THEOREM
If $V > -\infty$ and there exists $x \in \Omega$ with $g_i(x) < 0 \; \forall i$, then

$$\inf_{x \in \Omega : g(x) \le 0} f(x) = \sup_{\lambda \ge 0} \inf_{x \in \Omega} \Big(f(x) + \lambda^T g(x) \Big). \tag{L}$$

CONSEQUENCE

- Determine, for given $\lambda \ge 0$: $x_\lambda := \operatorname{argmin}_{x \in \Omega} \big(f(x) + \lambda^T g(x) \big)$.
- Insert x_λ in Lagrangian and determine $\lambda^* := \operatorname{argmax}_{\lambda \ge 0} \big(f(x_\lambda) + \lambda^T g(x_\lambda) \big)$.

Then $x^* := x_{\lambda^*}$ solves original problem.

[Right side of (1) is always bigger than left (show!). Equality needs existence of $x \in \Omega$ such that $g(x) < 0$.
The *complementary slackness* condition says that, either $\lambda_i^* = 0$ or the ith constraint is true with equality:

$$\lambda_i^* g_i(x^*) = 0, \qquad \text{for every } i = 1, \ldots, p.$$

]

Lagrange duality

The *Lagrangian* is the function $L\colon \Omega \times [0,\infty)^m \to \mathbb{R}$:

$$L(x,\lambda) = f(x) + \lambda^T g(x), \qquad\qquad g = (g_1,\ldots,g_m).$$

The Kuhn-Tucker theorem shows that the constraints $g(x) \le 0$ can be removed after replacing f by the Lagrangian.

If the unconstrained minimization $\inf_x L(x,\lambda)$ can be solved analytically, then one can next focus on the *dual problem*

$$\max_{\lambda \ge 0} L(x_\lambda, \lambda).$$

This is *concave problem* and sometimes easier.

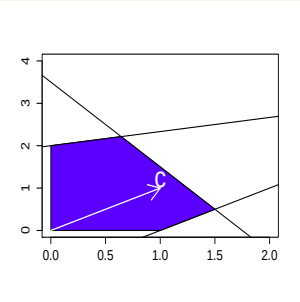
Concave functions

A function $f\colon \Omega \to \mathbb{R}$ is *concave* if $-f$ is convex.

Concave functions are nice for *maximization*.

Linear programming

Linear program



For given $c \in \mathbb{R}^n$, $(m \times n)$ matrix A and $b \in \mathbb{R}^m$:

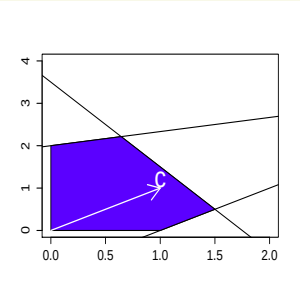
$$\max\Big\{c^T x: \qquad x \in \mathbb{R}^n, \quad x \geq 0, \quad Ax \leq b\Big\}.$$

If finite, the maximum is taken in a vertex of the polygon.

An *equality constraint* $Ax = b$ can be inserted as two inequality constraints $Ax \leq b$ and $(-A)x \leq -b$ (or directly).

[Inequalities are to be interpreted *coordinate-wise*: $x \geq 0$ means $x_i \geq 0$ for all i, and $Ax \leq b$ means $\sum_{j=1}^n A_{i,j} x_j \leq b_i$ for every i.]

Linear program — algorithms



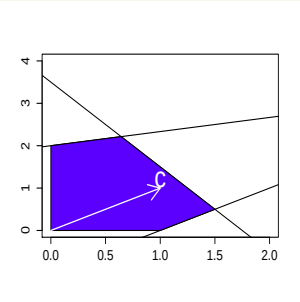
For given $c \in \mathbb{R}^n$, $(m \times n)$ matrix A and $b \in \mathbb{R}^m$:

$$\max\Big\{c^T x: \qquad x \in \mathbb{R}^n, \quad x \geq 0, \quad Ax \leq b\Big\}.$$

ALGORITHMS (many! popular in OR)

- Simplex method
- Interior point methods

Algorithms — R




```
> library(linprog)
> c=c(1,1)
> A=rbind(c(2,1),c(-1,3),c(1,-1))
> b=c(3.5,6,1)
> solveLP(c,b,A,TRUE)
Objective function (Maximum): 2.85714
       opt
1 0.642857
2 2.214286
--- Some output deleted-----
> lp("max",c,A,rep("<=",length(b)),b)$solution    # Equivalent (more stable?)
[1] 0.6428571 2.2142857
```

Example — median regression

Given $y \in \mathbb{R}^n$ and $(n \times p)$ matrix X find

$$\hat{\beta} = \operatorname*{argmin}_{\beta \in \mathbb{R}^p} \sum_{i=1}^{n} |y_i - (X\beta)_i|.$$

Example — median regression

Given $y \in \mathbb{R}^n$ and $(n \times p)$ matrix X find

$$\hat{\beta} = \operatorname*{argmin}_{\beta \in \mathbb{R}^p} \sum_{i=1}^{n} |y_i - (X\beta)_i|.$$

Write $y - X\beta = \mu - \nu$ for $\mu_i = \big(y_i - (X\beta)_i\big)^+$ and $\nu_i = \big(y_i - (X\beta)_i\big)^-$ and solve

$$\min\Big\{\sum_{i=1}^{n} \mu_i + \sum_{i=1}^{n} \nu_i \colon \mu \geq 0, \nu \geq 0, y - X\beta = \mu - \nu\Big\}.$$

Example — median regression

Given $y \in \mathbb{R}^n$ and $(n \times p)$ matrix X find

$$\hat{\beta} = \operatorname*{argmin}_{\beta \in \mathbb{R}^p} \sum_{i=1}^{n} |y_i - (X\beta)_i|.$$

Write $y - X\beta = \mu - \nu$ for $\mu_i = \big(y_i - (X\beta)_i\big)^+$ and $\nu_i = \big(y_i - (X\beta)_i\big)^-$ and solve

$$\min\Big\{\sum_{i=1}^{n} \mu_i + \sum_{i=1}^{n} \nu_i \colon \mu \geq 0, \nu \geq 0, y - X\beta = \mu - \nu\Big\}.$$

Rewrite further with $\beta = \beta_+ - \beta_-$ as

$$\min\Big\{\sum_{i=1}^{n} \mu_i + \sum_{i=1}^{n} \nu_i \colon \quad \beta_+ \geq 0, \beta_- \geq 0, \mu \geq 0, \nu \geq 0, X\beta_+ - X\beta_- + \mu - \nu = y\Big\}.$$

This is a linear program for $c^T x$ with equality constraints $Ax = b$ for

$$c = \begin{pmatrix} 0_p \\ 0_p \\ 1_n \\ 1_n \end{pmatrix}, \qquad x = \begin{pmatrix} \beta_+ \\ \beta_- \\ \mu \\ \nu \end{pmatrix}, \qquad A = \begin{pmatrix} X & -X & I_n & -I_n \end{pmatrix}, \qquad b = y.$$

Notation: plus and minus parts

For any number u we have

$$u = u^+ - u^-, \qquad |u| = u^+ + u^-,$$

for the plus and minus parts defined by

$$u^+ = \begin{cases} u & \text{if } u \geq 0 \\ 0 & \text{if } u < 0 \end{cases}, \qquad u^- = \begin{cases} 0 & \text{if } u \geq 0 \\ -u & \text{if } u < 0 \end{cases}.$$

The decomposition $u = u^+ - u^-$ is the decomposition of u as $u = a - b$ for nonnegative a, b with the minimal value of $a + b$.

Median regression through linear programming — R

```
> n=5; beta=1; x=runif(n); X=matrix(x,n,1); p=dim(X)[2]
> sigma=sample(c(1,3),n,replace=TRUE,prob=c(0.90,0.1)); y=beta*x+rnorm(n,0,sigma)
> library(quantreg);  rq(y~x-1)
       x
1.029661
> c=c(rep(0,2*p),rep(1,2*n))
> A=cbind(X,-X,diag(rep(1,n)),diag(rep(-1,n)))
> b=y
> round(cbind(A,b),1)
[1,] 0.6 -0.6 1 0 0 0 0 -1  0  0  0  0 -1.1
[2,] 0.3 -0.3 0 1 0 0 0  0 -1  0  0  0 -1.3
[3,] 0.8 -0.8 0 0 1 0 0  0  0 -1  0  0  1.3
[4,] 0.0  0.0 0 0 0 1 0  0  0  0 -1  0 -0.3
[5,] 0.8 -0.8 0 0 0 0 1  0  0  0  0 -1  0.9
>  solveLP(c,b,A,const.dir=rep("=",n),lpSolve=TRUE)
Objective function (Minimum): 4.09239
        opt
1  1.029661
2  0.000000
3  0.000000
4  0.000000
5  0.402486
6  0.000000
7  0.000000
8  1.713898
9  1.605126
10 0.000000
11 0.370885
12 0.000000
--- Some output deleted-----
> lp("min",c,A,rep("=",n),b)$solution    # Equivalent (more stable?) function
 [1] 1.0296614 0.0000000 0.0000000 0.0000000 0.4024862 0.0000000 0.0000000 1.7138977 1.6051259 0.0000000
[11] 0.3708847 0.0000000
```

[Alternative to equality constraints: `AA=rbind(A,-A); bb=c(y,-y)` and now `solveLP(c,bb,AA,FALSE,lpSolve=TRUE)` . Careful: without `lpSolve=TRUE` it does not work (bug!).]

Quadratic programming

Quadratic program

For given positive-definite $n \times n$ matrix D, $d \in \mathbb{R}^n$, $(n \times m)$ matrix A and $b \in \mathbb{R}^m$:

$$\min\left\{\tfrac{1}{2}x^T Dx - d^T x: \qquad x \in \mathbb{R}^n, \quad A^T x \geq b\right\}.$$

An *equality constraint* $Ax = b$ can be inserted as two inequality constraints $Ax \leq b$ and $(-A)x \leq -b$ (or directly).

The criterion function is convex and strictly convex if D is nonsingular.

ALGORITHMS:

- Many

[D positive-defined means that $x^T Dx \geq 0$ for all x.]

Algorithms — R

```
> library(quadprog)
> c=c(1,1)
> A=rbind(c(2,1),c(-1,3),c(1,-1))
> b=c(3.5,6,1)
> lp("max",c,A,rep("<=",length(b)),b)$solution
[1] 0.6428571 2.2142857
> AQ=cbind(-t(A),diag(c(1,1))); bQ=c(-b,c(0,0))
> D=diag(0.000001*c(1,1))
> solve.QP(D,c,AQ,bQ)
$solution
[1] 0.6428571 2.2142857
$value
[1] -2.85714
---- Some output deleted----
```

[The function `solve.QP` expects $D > 0$ (strict), constraints $Ax \geq b$, and does not impose $x \geq 0$.]

Example — LASSO

In linear regression problem $Y = X\beta + e$ estimate β by

$$\operatorname*{argmin}_{\beta} \Big[\|Y - X\beta\|^2 + \lambda \sum_{i=1}^{p} |\beta_i| \Big].$$

Rewrite as

$$\begin{aligned} &Y^T Y - 2Y^T X(\beta_+ - \beta_-) + (\beta_+ - \beta_-)^T X^T X(\beta_+ - \beta_-) + \lambda 1_p^T(\beta_+ + \beta_-) \\ &\quad = (\beta_+^T, \beta_-^T) \begin{pmatrix} X^T X & -X^T X \\ -X^T X & X^T X \end{pmatrix} \begin{pmatrix} \beta_+ \\ \beta_- \end{pmatrix} + (-2X^T Y + \lambda 1_p)^T \beta_+ + (2X^T Y + \lambda 1_p)^T \beta_-. \end{aligned}$$

Quadratic program with constraints $\beta_+ \geq 0$ and $\beta_- \geq 0$.

[The LARS algorithm is written especially for this problem, and is preferable over general purpose QP algorithms]

Example — support vector machine

This is a classification algorithm using a hyperplane: given training data $(x_1, y_1), \ldots, (x_n, y_n)$ with $y_i \in \{-1, 1\}$ it predicts the label y for a new x based on which side of the hyperplane x falls.

The hyperplane is determined to maximize the minimum distance of the training data to the hyperplane. It is calculated by a quadratic program.