# Convex Optimisation

## Lecture 1

Optimisation problems in general:

* Minimise f\_0(x)
* Subject to:
* f\_i(x) >= 0 for i=1,…,m
* g\_j(x) = 0 for j=1,…,n

for x in R^n.

*Convex* optimisation:

* Minimise f\_0(x)
* Subject to:
* f\_i(x) >= 0 for i=1,…,m
* Ax=b

Need the equality constrains (g\_j in the general case) to be linear.

f\_0,…,f\_m (i.e. the objective and the inequality constraints) must all be convex functions:

* i.e. for t in [0,1]: f\_i(tx + (1-t)y) <= t\*f\_i(x) + (1-t)\*f\_i(y)
* i.e. f\_i have non-negative curvature

Convex Optimisation – Conic Form

* Min (c^T)x
* Subject to: Ax = b, x in K
* Where x in R^n, and K is convex cone

This is the canonical form of convex optimisation problems; special cases include linear programming, semidefinite programming.

This is used as the intermediate form by computer solvers.

Convexity

f is concave is -f is convex

Convex optimisation minimises convex functions, or maximises concave function

f is affine if it is both convex and concave – i.e. it is linear (of form Ax + b)

Most functions are neither convex nor concave globally, e.g. consider sin(x) for x in R

Can verify function is convex by checking Hessian is positive semi-definite.

But this is difficult for more complicated (high-dim) functions. Instead, have a library of convex/concave basic functions; then have a set of transformations that preserve convexity.

e.g. Basic convex functions (not all these are obvious; proofs omitted):

* x^2/y for y>0 is convex in (x,y); also “marginally convex” – varying only x or only y also convex, as just get scaled versions of x^2 and 1/y resp. But note, marginal convexity in all variables doesn’t imply joint-convexity over the high-dimensional space (does the converse hold? Probably…).
* Same is true for (x^T)x/y for x in R^n, y>0. And for (x^T)(Y^-1)x for Y nxn positive definite matrix, x in R^n.
* log(e^(x1) + … + e^(xn)) is convex – i.e the softmax
* Sum of k largest entries in some vector x in R^n (k<=n). e.g. k=1 – max(x1,…xn)
* xlog(x/y) is convex for x,y > 0 (e.g. consider KL divergence: plog(p/q))
* Max eigenvalue of a symmetric matrix

e.g. Basic concave functions:

* log(det(X)) for X positive-definite; similarly det(X)^(1/n). This is the entropy of a Gaussian rv with covariance X
* log of 1D Gaussian CDF
* Min eigenvalue of symmetric matrix

Convex Calculus

For convex function f, the following transformations preserve convexity:

* Non-negative Scaling: a\*f for scalar a>=0
* Summation: f+g for convex function g
* Affine pre-composition: f(Ax + b)
* Pointwise max: f\_1,…,f\_m convex => max{f\_i(x)} over i
* Composition: h(f(x) for h convex and increasing

This can be used to show more complicate functions are convex, e.g.:

* Sum-of-squares cost with L1 regularisation: ||Ax – b||\_2 + lambda \* ||x||\_1 for lambda >= 0
* KL Divergence on two non-negative vectors: D(u,v) = sum of u\_i\*log(u\_i/v\_i) – u\_i + v\_i, over i=1,…,n, for u,v >0
* log[{e^(a\*f(x,w1)) + … + e^(a\*f(x,wk))}/k], which is log{E[e^(a\*f(x,w))]} for w ~ U({w1, …, wk}) i.e. uniform rv on {w1,…,wk}, is convex. This logE comes up as the following upper bound:
  + log{P(f(x,w) >= 0)} <= log{E[e^(a\*f(x,w))]}

The rules above are all just special cases of the single rule:

* h(f\_1(x), …, f\_k(x)) is convex when h is convex and, for each i:
  + h is increasing in argument i, and f\_i is convex; or
  + h is decreasing in argument i, and f\_i is concave; or
  + f\_i is affine

Note: the rules are sufficient but not necessary.

Disciplined Convex Program

DCP’s are of form:

* Zero or one **objective**, of form:
  + Min {scalar convex expression}; or
  + Max {scalar concave expression}
* Subject to zero or more **constraints**, of form:
  + {convex expression} <= {concave expression}; or
  + {concave expression} >= {convex expression}; or
  + {affine expression} = {affine expression}