STAT 5361: Statistical Computing

Jun Yan

Department of Statistics
University of Connecticut

Outline I

1. Non-Stochastic Optimization
   - Some Background
   - Univariate Problems
   - Multivariate Problems
2. Combinatorial Optimization
   - Local Search
   - Simulated Annealing
   - Genetic Algorithm
3. EM and MM Algorithms
   - EM Algorithm
   - Majorization-Minimization Algorithm
     - Example: Penalized AFT model with induced smoothing
4. Simulation Basics
   - Inverse Transform Method
   - Rejection Sampling
   - Tranformation
   - Variants of Rejection Sampling

Outline II

- Sampling importance resampling
- Sampling using Markov Chains

5 Simulation of Stochastic Processes

- Simulating Brownian motions
- Gaussian short rate models
- Processes with Jumps

Non-stochastic Optimization

What we will learn:

- Some basics about statistical inference
- Univariate problems
  - Newton's method
  - Fisher scoring
  - Secant method
  - Fixed-point method
  - Connections of the above
- Multivariate problems
  - Newton's method
  - Newton-like methods

Background: likelihood inference

- Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ be an i.i.d. sample from

$$f(\boldsymbol{x} \,|\, \boldsymbol{\theta}^*),$$

  with the true parameter value $\boldsymbol{\theta}^*$ being unknown.

- The likelihood function is

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{n} f(\boldsymbol{x}_i \,|\, \boldsymbol{\theta}),$$

- The *maximum likelihood estimator* (MLE) of the parameter value is the maximizer of $L(\boldsymbol{\theta})$. MLE has the invariate property.
- Usually it is easier to work with the *log likelihood function*

$$l(\boldsymbol{\theta}) = \log L(\boldsymbol{\theta}).$$

- Typically, maximization of $l(\boldsymbol{\theta})$ is done by solving

$$l'(\boldsymbol{\theta}) = 0.$$

  - $l'(\boldsymbol{\theta})$ is called the *score function*.
  - For each possible parameter value $\boldsymbol{\theta}$, $l(\boldsymbol{\theta})$ is a random variable, because it depends on the observed values of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$.

- For any $\boldsymbol{\theta}$,

$$\mathbb{E}_{\boldsymbol{\theta}} \left\{ l'(\boldsymbol{\theta}) \right\} = 0,$$
$$\mathbb{E}_{\boldsymbol{\theta}} \left\{ l'(\boldsymbol{\theta}) l'(\boldsymbol{\theta})^T \right\} = -\mathbb{E}_{\boldsymbol{\theta}} \left\{ l''(\boldsymbol{\theta}) \right\}.$$

  where $\mathbb{E}_{\boldsymbol{\theta}}$ is the expectation with respect to $f(\boldsymbol{x} \,|\, \boldsymbol{\theta})$. The equality holds true under mild regularity conditions.

-

$$I(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\theta}} \left\{ l'(\boldsymbol{\theta}) l'(\boldsymbol{\theta})^T \right\}$$

  is known as the *Fisher information*.

- The importance of the Fisher information $I(\boldsymbol{\theta})$ is that it sets the limit on how accurate an unbiased estimate of $\boldsymbol{\theta}$ can be.
- As $n \to \infty$, the asymptotic distribution of $\sqrt{n}(\widehat{\boldsymbol{\theta}}_{\text{MLE}} - \boldsymbol{\theta}^*)$ is $N_p(\mathbf{0}, nI(\boldsymbol{\theta}^*)^{-1})$. Since θ^* is unknown, the asymptotic covariance matrix $I(\boldsymbol{\theta}^*)^{-1})$ needs to be estimated.
  - If $\dim(\boldsymbol{\theta}) = 1$, $I(\boldsymbol{\theta})$ is a nonnegative number.
  - If $\dim(\boldsymbol{\theta}) > 1$, $I(\boldsymbol{\theta})$ is a nonnegative definite matrix.
- The *observed Fisher information* is

$$-l''(\boldsymbol{\theta}).$$

The expected Fisher information $I(\boldsymbol{\theta})$ may not be easily computed. The observed Fisher information is a good approximation to $I(\boldsymbol{\theta})$ that improves as n gets bigger.

Besides MLEs, there are other likelihoods for parameter estimation. Suppose $\boldsymbol{\theta}$ has two parts: $\boldsymbol{\theta} = (\phi, \boldsymbol{\mu})$ and we are only interested in ϕ. The *profile likelihood* for ϕ is

$$L(\phi) := \max_{\boldsymbol{\mu}} L(\phi, \boldsymbol{\mu}).$$

- $\boldsymbol{\mu}$ is the nuisance parameter.
- Need to maximize $L(\phi, \boldsymbol{\mu})$ for every fixed ϕ. Also need to maximize $L(\phi)$.

A general method

Suppose $g(\boldsymbol{x})$ is a differentiable function, where

$$\boldsymbol{x} = (x_1, \dots, x_n).$$

To find its maximum (or minimum), one method is to solve the equation

$$g'(\boldsymbol{x}) = 0$$
$$\text{where} \quad g'(\boldsymbol{x}) = \left(\frac{\partial g}{\partial x_1}, \dots, \frac{\partial g}{\partial x_n} \right)^T.$$

Then the maximization is equivalent to solving

$$f(\boldsymbol{x}) = 0,$$

where $f = g'$.
For maximum likelihood estimation, g is the log likelihood function l, and $\boldsymbol{x}$ is the corresponding parameter vector $\boldsymbol{\theta}$.

Univariate Problems

Optimization: Univariate case

- Goal: optimize a real-valued function g with respect to its argument, a p dimensional vector $\boldsymbol{x}$. We will first consider the case $p = 1$.
- We will limit consideration mainly to smooth and differentiable functions.
- Root finding methods. Solving unconstrained nonlinear equations.
- Iterative algorithms: starting value, updating equation, stopping rule/convergence criterion.

Using bisection method to maximize

$$g(x) = \frac{\log x}{1 + x}.$$

or to sovle

$$f(x) = g'(x) = \frac{1 + \frac{1}{x} - \log x}{(1 + x)^2} = 0.$$

Newton's method

This is a fast approach to finding roots of a differentiable function $f(x)$. First, set initial value x_0. Then for $t = 0, 1, \ldots$, compute

$$x_{t+1} = x_t + h_t, \text{ with } h_t = -\frac{f(x_t)}{f'(x_t)}.$$

Continue the iterations until x_t converges.

- Also known as Newton-Raphson iteration.
- Need to specify x_0.
- If $f(x) = 0$ has multiple solutions, the end result depends on x_0.

Newton's method require computing the derivative of a function. Algorithms are available to find root(s) of a univariate function without having to compute its derivative. For example, in R, one can use `uniroot`. Newton's method can be applied to optimize g by applying to

$$f = g'.$$

- Both g' (*gradient*) and g'' (*Hessian*) are needed.
- Many variants of Newton's method avoid the computation of the Hessian, which can be difficult especially for multivariate functions.

To maximize

$$g(x) = \frac{\log x}{1+x},$$

first find

$$f(x) = g'(x) = \frac{1 + \frac{1}{x} - \log x}{(1+x)^2},$$

$$f'(x) = g''(x) = -\frac{3 + 4/x + 1/x^2 - 2\log x}{(1+x)^3}.$$

So in the Newton's method,

$$h_t = \frac{(x_t + 1)(1 + 1/x_t - \log x_t)}{3 + 4/x_t + 1/x_t^2 - 2\log x_t}.$$

Note that to solve $f(x) = 0$, one can instead solve $1 + 1/x - \log x = 0$. Treat this as a new f function. Then in the Newton's method,

$$h_t = x_t - \frac{x_t^2 \log x_t}{1 + x_t} \quad \Longrightarrow \quad x_{t+1} = 2x_t - \frac{x_t^2 \log x_t}{1 + x_t}.$$

To maximize the log likelihood $l(\theta)$, Newton's method computes

$$\theta_{t+1} = \theta_t - \frac{l'(\theta_t)}{l''(\theta_t)}$$

Example: consider $x_1, \ldots, x_n \sim$ i.i.d. $N(\mu, \sigma^2)$.

Example: consider the model on shift

$$p(x \mid \theta) = p(x - \theta).$$

Given observations $x_1, \ldots, x_n$ i.i.d. $\sim p(x \mid \theta)$

$$\begin{aligned} l(\theta) &= \sum_{i=1}^n \log p(x_i - \theta) \\ l'(\theta) &= -\sum_{i=1}^n \frac{p'(x_i - \theta)}{p(x_i - \theta)} \\ l''(\theta) &= \sum_{i=1}^n \frac{p''(x_i - \theta)}{p(x_i - \theta)} - \sum_{i=1}^n \left\{ \frac{p'(x_i - \theta)}{p(x_i - \theta)} \right\}^2 . \end{aligned}$$

Note that we need to update θ in the iterations, not $x_1, \ldots, x_n$.

In R, to *minimize* a function, one can use

```
z = nlminb(x0, g, gr.g, hess.g)
```

here `x0` is the initial value, g is the function being minimized, `gr.g` its gradient and `hess.g` its Hessian. (Unconstrained and box-constrained optimization using PORT routines).
In the above function, the derivatives g' and g'' have to be analytically calculated. One can also use

```
z = nlminb(x0, g, gr.g)
```

without inputing the analytic expression of g'', or, even simpler,

```
z = nlminb(x0, g)
```

without inputing the analytic expressions of either derivatives. In these cases, numerical approximations of derivatives will be computed during the iterations.
Other functions for optimization in R include `optim`, `optimize`, `nlm` and `constrOptim`.

For profile likelihood

$$L(\phi) = \max_{\boldsymbol{\mu}} L(\phi, \boldsymbol{\mu})$$

we need to optimize $L(\phi, \boldsymbol{\mu})$ for each given ϕ.
In R, in order to get $\min_x g(x, y)$ for each fixed y, one can do the following. First, define `g(x,y)`, `gr.g(x,y)`, etc. Then call, say,

```
nlminb(x0, g, gr.g, hess.g, y=1),   or
nlminb(x0, g, gr.g, y=.3),   or
nlminb(x0, g, y=-1)
```

If one only wants minimization for $x \in [a, b]$, use,

```
nlminb(x0, ..., lower=a, upper=b)
```

Fisher scoring

This is a variant of Newton's method specific for MLE. Recall $-l''(\theta)$ is the observed Fisher information at θ. To maximize $l(\theta)$, an alternative is to replace $-l''(\theta_t)$ with $I(\theta_t)$ to get

$$\theta_{t+1} = \theta_t + \frac{l'(\theta_t)}{I(\theta_t)}.$$

To *minimize* $-l(\theta)$, one still can use

```
z = nlminb(x0, f, grf, fs)
```

where `f` is $-l(\theta)$, `grf` is $-l'(\theta)$, and `fs` is $I(\theta)$ instead of $-l''(\theta)$. Generally, use Fisher scoring in the beginning to make rapid improvements, and Newton's method for refinement near the end.

Continuing the example on $p(x \mid \theta) = p(x - \theta)$, to use Fisher scoring, we need to compute

$$I(\theta) = -\mathbb{E}_\theta[l''(\theta)].$$

We already know

$$l''(\theta) = \sum_{i=1}^n \frac{p''(x_i - \theta)}{p(x_i - \theta)} - \sum_{i=1}^n \left\{ \frac{p'(x_i - \theta)}{p(x_i - \theta)} \right\}^2 .$$

Therefore

$$I(\theta) = -n\mathbb{E}_\theta \left[\frac{p''(X - \theta)}{p(X - \theta)} - \left\{ \frac{p'(X - \theta)}{p(X - \theta)} \right\}^2 \right].$$

Since under parameter θ, X has density $p(x-\theta)$, the last $\mathbb{E}_\theta[...]$ equals

$$\begin{aligned}
&\int \left[\frac{p''(x-\theta)}{p(x-\theta)} - \left\{ \frac{p'(x-\theta)}{p(x-\theta)} \right\}^2 \right] p(x-\theta)\, \mathrm{d}x \\
&= \int p''(x-\theta)\, \mathrm{d}x - \int \frac{[p'(x-\theta)]^2}{p(x-\theta)}\, \mathrm{d}x \\
&= \frac{\mathrm{d}^2}{\mathrm{d}\theta^2} \int p(x-\theta)\, \mathrm{d}x - \int \frac{\{p'(x)\}^2}{p(x)}\, \mathrm{d}x \\
&= \frac{\mathrm{d}^2}{\mathrm{d}\theta^2} 1 - \int \frac{\{p'(x)\}^2}{p(x)}\, \mathrm{d}x = -\int \frac{\{p'(x)\}^2}{p(x)}\, \mathrm{d}x.
\end{aligned}$$

Therefore,

$$I(\theta) = n \int \frac{\{p'(x)\}^2}{p(x)}\, \mathrm{d}x.$$

So, in this case, Fisher information is a constant.

Secant method

Approximating $f'(x_t)$ by

$$\frac{f(x_t) - f(x_{t-1})}{x_t - x_{t-1}},$$

the Newton's method turns into the secant method

$$x_{t+1} = x_t - \frac{f(x_t)(x_t - x_{t-1})}{f(x_t) - f(x_{t-1})}.$$

- Need to specify x_0 and x_1.

Fixed point iteration

Compute

$$x_{t+1} = x_t + \alpha f(x_t), \quad t = 0, 1, \dots,$$

where $\alpha \neq 0$ is a tuning parameter so that

$$|1 + \alpha f'(x)| \le \lambda < 1, \text{ all } x$$

or more generally,

$$|x - y + \alpha[f(x) - f(y)]| \le \lambda |x - y|, \qquad \text{any } x, y$$

with $0 \le \lambda < 1$ a constant, i.e., $F(x) = x + \alpha f(x)$ is a *contraction*.
If such α exists, the speed of convergence depends on λ. The smaller λ is, the faster the convergence.

Some Details on Fixed point iteration

Definition: A fixed point of a function is a point whose evaluation by that function equals to itself, i.e., $x = G(x)$.
Fixed point iteration: the natural way of hunting for a fixed point is to use $x_{t+1} = G(x_t)$.
Definition: A function G is contractive on $[a, b]$ if

(1). $G(x) \in [a, b]$ whenever $x \in [a, b]$,
(2). $|G(x_1) - G(x_2)| \leq \lambda |x_1 - x_2|$ for all $x_1, x_2 \in [a, b]$ and some $\lambda \in [0, 1)$.

Theorem: if G is contractive on $[a, b]$, then there is a unique fixed point $x^* \in [a, b]$, and the fixed point iteration convergence to it when starting inside the interval.
Convergence:
$|x_{t+1} - x_t| = |G(x_t) - G(x_{t-1})| \leq \lambda |x_t - x_{t-1}| \leq \lambda^t |x_1 - x_0| \to 0$, as $t \to \infty$. It follows that $\{x_t\}$ convergent to a limit x^*.

Connection between fixed-point method and Newton methods

Root-finding problems using fixed point iteration: for solving $f(x) = 0$, we can simply let $G(x) = x + \alpha f(x)$, where $\alpha \neq 0$ is a constant.
Required Lipschitz condition: $|x - y + \alpha[f(x) - f(y)]| \leq \lambda |x - y|$, for some $\lambda \in [0, 1)$ and for all $x, y \in [a, b]$. This holds if $|G'(x)| \leq \lambda$ for some $\lambda \in [0, 1)$ and for all $x \in [a, b]$, i.e., $|1 + \alpha f'(x)| \leq \lambda$. (use mean value theorem.)
Newton methods: $G(x) = x - f(x)/f'(x)$. So it is as if α_t is chosen adaptively as $\alpha_t = -1/f'(x_t)$. This leads to a faster convergence order (quadratic).

Convergence Order

Define $\varepsilon_t = x_t - x^*$. A method has convergence order β if

$$\lim_{t\to\infty} \varepsilon_t = 0, \qquad \lim_{t\to\infty} \frac{|\varepsilon_{t+1}|}{|\varepsilon_t|^\beta} = c,$$

for some constant $c \neq 0$ and $\beta > 0$.

- For Newton's method, $\beta = 2$. (If it converges.)
- For Secant method, $\beta \approx 1.62$.
- For fixed point iteration, $\beta = 1$.

Convergence Order

For Newton's method: suppose f has two continuous derivatives and $f'(x^*) \neq 0$. There then exists a neighborhood of x^* within which $f'(x) \neq 0$ for all x. By Taylor expansion,

$$0 = f(x^*) = f(x_t) + f'(x_t)(x^* - x_t) + \frac{1}{2}f''(q)(x^* - x_t)^2,$$

for some q between x^* and x_t. Rearranging terms, we find that

$$\frac{\varepsilon_{t+1}}{\varepsilon_t^2} = \frac{f''(q)}{2f'(x_t)} \to \frac{f''(x^*)}{2f'(x^*)}.$$

Convergence Order

For fixed point iteration: let G be a continuous function on the closed interval $[a, b]$ with $G : [a, b] \to [a, b]$ and suppose that G' is continuous on the open interval (a, b) with $|G'(x)| \leq k < 1$ for all $x \in (a, b)$. If $g'(x^*) \neq 0$, then for any $x_0 \in [a, b]$, the fixed point iteration converges linearly to the fixed point x^*. This is because

$$|x_{t+1} - x^*| = |G(x_t) - G(x^*)| = |G'(q)||x_t - x^*|,$$

for some q between x_t and x^*. This implies that

$$\frac{|\varepsilon_{t+1}|}{|\varepsilon_t|} \to |G'(x^*)|.$$

Stopping Rules

- Absolute convergence criterion:

$$\|x_{t+1} - x_t\| < \epsilon,$$

  where ϵ is a chosen tolerance.

- Relative convergence criterion

$$\frac{\|x_{t+1} - x_t\|}{\|x_t\|} < \epsilon.$$

  If x_t close to zero,

$$\frac{\|x_{t+1} - x_t\|}{\|x_t\| + \epsilon} < \epsilon.$$

- Many other rules depending on the algorithm.

Multivariate Problems

Newton's method

Let g now be a function in $\boldsymbol{x} = (x_1, \dots, x_p)^T$.
The generalization is straightforward: to maximize $g(\boldsymbol{x})$, set

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - [g''(\boldsymbol{x}_t)]^{-1} g'(\boldsymbol{x}_t).$$

- $g''(\boldsymbol{x})$ is a $p \times p$ matrix with the (i, j)th entry equal to

$$\frac{\partial^2 g(\boldsymbol{x})}{\partial x_i \partial x_j};$$

- $g'(\boldsymbol{x})$ is a $p \times 1$ vector with the ith entry equal to

$$\frac{\partial g(\boldsymbol{x})}{\partial x_i}.$$

Newton-like methods

For MLE, the generalization is straightforward. Simply use

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + I(\boldsymbol{\theta}_t)^{-1} l'(\boldsymbol{\theta}_t).$$

These methods in each iteration approximate $g''(\boldsymbol{x}_t)$ by some $p \times p$ matrix $\boldsymbol{M}_t = \boldsymbol{M}_t(\boldsymbol{x}_t)$ to get

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \boldsymbol{M}_t^{-1} g'(\boldsymbol{x}_t). \tag{1}$$

Of course, $\boldsymbol{M}_t$ should be easier to compute than $g''(\boldsymbol{x}_t)$.
Fisher scoring is a Newton-like method, because it uses

$$\boldsymbol{M}_t = -I(\boldsymbol{\theta}_t)$$

in place of $-l''(\boldsymbol{\theta}_t)$.

Steepest ascent methods

In (1), set

$$\boldsymbol{M}_t = -\alpha_t^{-1}\boldsymbol{I}_p,$$

so that

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \alpha_t g'(\boldsymbol{x}_t),$$

where $\alpha_t > 0$ is the step size at t which can shrink to ensure ascent. If at step t, the original step turns out to be downhill, the updating can backtrack by halving α_t.

Discrete Newton and Fixed-point methods

In fixed-point methods, usually $\boldsymbol{M}_t$ is fixed to be $\boldsymbol{M}$, e.g., $g''(\boldsymbol{x}_0)$. This amounts to applying univariate scaled fixed-point iteration to each component.

In discrete Newton methods, $g''(\boldsymbol{x}_t)$ is approximated as follows. Compute matrix $\boldsymbol{M}_t$ with the $(i, j)^{\text{th}}$ entry equal to

$$\boldsymbol{M}_t(i, j) = \frac{g_i'(\boldsymbol{x}_t + h_t(i, j)\boldsymbol{e}_j) - g_i'(\boldsymbol{x}_t)}{h_t(i, j)}$$

for some constant $h_t(i, j) \neq 0$, where

$$g_i'(\boldsymbol{x}) = \frac{\partial g(\boldsymbol{x})}{\partial x_i}.$$

If $h_t(i, j)$ is small, then

$$\boldsymbol{M}_t(i, j) \approx \frac{\partial^2 g(\boldsymbol{x}_t)}{\partial x_i \partial x_j}$$

and so $\boldsymbol{M}_t$ approximates $g''(\boldsymbol{x}_t)$.

However, since $g''(\boldsymbol{x}_t)$ is symmetric, instead of using $\boldsymbol{M}_t$ directly, use the symmetric

$$(\boldsymbol{M}_t + \boldsymbol{M}_t^T)/2$$

as the approximation of $g''(\boldsymbol{x}_t)$.

- No need to calculate second order derivatives
- Inefficient: all p^2 entries have to be updated each time.

Quasi-Newton methods

These methods aim to achieve the following goals.

- Each step satisfies the secant condition

$$g'(\boldsymbol{x}_{t+1}) - g'(\boldsymbol{x}_t) = \boldsymbol{M}_{t+1}(\boldsymbol{x}_{t+1} - \boldsymbol{x}_t).$$

- No need to compute second order derivatives.
- Maintain symmetry of $\boldsymbol{M}_t$.
- Aim to update $\boldsymbol{M}_t$ efficiently.

There is a unique rank-one method that satisfies the secant condition and maintains the symmetry of $\boldsymbol{M}_t$: after getting

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \boldsymbol{M}_t^{-1} g'(\boldsymbol{x}_t)$$

compute

$$\begin{aligned} \boldsymbol{z}_t &= \boldsymbol{x}_{t+1} - \boldsymbol{x}_t, \quad \boldsymbol{y}_t = g'(\boldsymbol{x}_{t+1}) - g'(\boldsymbol{x}_t), \\ \boldsymbol{v}_t &= \boldsymbol{y}_t - \boldsymbol{M}_t \boldsymbol{z}_t, \quad c_t = \frac{1}{\boldsymbol{v}_t^T \boldsymbol{z}_t}. \end{aligned}$$

Then update

$$\boldsymbol{M}_{t+1} = \boldsymbol{M}_t + c_t \boldsymbol{v}_t \boldsymbol{v}_t^T.$$

Note $\boldsymbol{M}_{t+1} - \boldsymbol{M}_t$ is of rank one. We can verify the secant condition is satisfied by multiplying both sides by $\boldsymbol{z}_t$.

There are several rank-two method satisfying the secant condition and the symmetry requirement. The Broyden class updates $\boldsymbol{M}_t$ as follows. After getting $\boldsymbol{x}_{t+1}$ and $\boldsymbol{z}_t$, $\boldsymbol{y}_t$ as above, compute

$$\boldsymbol{d}_t = \frac{\boldsymbol{y}_t}{\boldsymbol{z}_t^T \boldsymbol{y}_t} - \frac{\boldsymbol{M}_t \boldsymbol{z}_t}{\boldsymbol{z}_t^T \boldsymbol{M}_t \boldsymbol{z}_t}.$$

Then update

$$\boldsymbol{M}_{t+1} = \boldsymbol{M}_t - \frac{\boldsymbol{M}_t \boldsymbol{z}_t (\boldsymbol{M}_t \boldsymbol{z}_t)^T}{\boldsymbol{z}_t^T \boldsymbol{M}_t \boldsymbol{z}_t} + \frac{\boldsymbol{y}_t \boldsymbol{y}_t^T}{\boldsymbol{z}_t^T \boldsymbol{y}_t} + \delta_t (\boldsymbol{z}_t^T \boldsymbol{M}_t \boldsymbol{z}_t) \boldsymbol{d}_t \boldsymbol{d}_t^T,$$

where δ_t is a parameter.

A popular method to solve unconstrained nonlinear optimization problems is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, with $\delta_t \equiv 0$.

In R, `optim` can be called to minimize a function, using the BFGS method or its variant, the “L-BFGS-B” method. “L” stands for limited memory, and “B” stands for box constraint.

Approximating Hessian for MLE

For MLE, the Hessian $l''(\widehat{\boldsymbol{\theta}}_{\text{MLE}})$ is critical because it provides estimates of standard error and covariance.

- Quasi-Newton methods may provide poor approximation because it is based on the idea of using poor approximation of the Hessian to find the root for $l'(\boldsymbol{\theta}) = 0$.
- Use the discrete multivariate Newton method with

$$\boldsymbol{M}_t(i,j) = \frac{l_i'(\boldsymbol{\theta}_t + h_{ij}\boldsymbol{e}_j) - l_i'(\boldsymbol{\theta}_t - h_{ij}\boldsymbol{e}_j)}{2h_{ij}}.$$

Gauss-Newton method

Example: nonlinear regression. In many cases, we want to maximize

$$g(\boldsymbol{\theta}) = -\sum_{i=1}^{n} (y_i - f_i(\boldsymbol{\theta}))^2,$$

where each $f_i(\boldsymbol{\theta})$ is differentiable. Recall that for linear regression:

$$y_i = \boldsymbol{x}_i^T \boldsymbol{\theta} + \varepsilon, \quad i = 1, \ldots, n$$

the LS estimator of $\boldsymbol{\theta}$ maximizes $g(\boldsymbol{\theta})$ with $f_i(\boldsymbol{\theta}) = \boldsymbol{x}_i^T \boldsymbol{\theta}$ and equals

$$\widehat{\boldsymbol{\theta}} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{y}$$

where

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x}_1^T \\ \vdots \\ \boldsymbol{x}_n^T \end{pmatrix}, \quad \boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}.$$

The Gauss-Newton method applies a similar idea to the nonlinear case. Let $\boldsymbol{\theta}^*$ be the (unknown) maximizer of $g(\boldsymbol{\theta})$. Given any candidate $\boldsymbol{\theta}$, the function

$$h(\boldsymbol{u}) = -\sum_{i=1}^{n} (y_i - f_i(\boldsymbol{\theta} + \boldsymbol{u}))^2.$$

is maximized by $\boldsymbol{\beta} = \boldsymbol{\theta}^* - \boldsymbol{\theta}$. Of course, $\boldsymbol{\beta}$ is unknown. However, if $\boldsymbol{\theta}$ is near $\boldsymbol{\theta}^*$, then $\boldsymbol{\beta} \approx 0$, so by Taylor expansion of $h(\boldsymbol{u})$, it may be close to the maximizer of

$$-\sum_{i=1}^{n} (y_i - f_i(\boldsymbol{\theta}) - f_i'(\boldsymbol{\theta})^T \boldsymbol{u})^2.$$

Treat $y_i - f_i(\boldsymbol{\theta})$ the same way as y_i in the linear regression, and $f_i'(\boldsymbol{\theta})$ the same way as $\boldsymbol{x}_i$, then

$$\boldsymbol{\theta}^* - \boldsymbol{\theta} \approx (\boldsymbol{A}^T \boldsymbol{A})^{-1} \boldsymbol{A}^T \boldsymbol{z}$$

where

$$\boldsymbol{z} = \boldsymbol{z}(\boldsymbol{\theta}) = \begin{pmatrix} y_1 - f_1(\boldsymbol{\theta}) \\ \vdots \\ y_n - f_n(\boldsymbol{\theta}) \end{pmatrix}, \quad \boldsymbol{A} = \boldsymbol{A}(\boldsymbol{\theta}) = \begin{pmatrix} f_1'(\boldsymbol{\theta})^T \\ \vdots \\ f_n'(\boldsymbol{\theta})^T \end{pmatrix}.$$

The update rule thus is

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + (\boldsymbol{A}_t^T \boldsymbol{A}_t)^{-1} \boldsymbol{A}_t^T \boldsymbol{z}_t,$$

where $\boldsymbol{z}_t = \boldsymbol{z}(\boldsymbol{\theta}_t)$ and $\boldsymbol{A}_t = \boldsymbol{A}(\boldsymbol{\theta}_t)$.
In R, the Gauss-Newton method is the default method of the function `nls`.

Practical Issues

- Initial value
- Convergence criteria: based on a distance measures for vectors
- Multiple local optima
  - Multiple initial values
  - Compare the objective function value at convergences
- Trade-off between efficiency and robustness

Combinatorial Optimization

Motivation:

- There are hard optimization problems for which most methods including what we have discussed so far are useless.
- These problems are usually combinatorial in nature. Maximization requires a discrete search of a very large space.
- Problem: maximize $f(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_p)$, where $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ and $\boldsymbol{\Theta}$ consists of N elements.
  - Each $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ is called a candidate solution.
  - Usually N is a very large number depending on problem size p.
  - The difficulty of a particular size-p problem can be characterized by the number of operations required to solve it in the worst case scenario using the best known algorithm.
  - Suppose a problem is $\mathcal{O}(p!)$. If it requires 1 minute to solve for $p = 20$, it would take 12.1 years for $p = 25$ and 207 million years for $p = 30$.

What can we do:

- We have to take some sacrifices: we will abandon the global algorithms and focus on algorithms that can find a good local solution.
- Heuristic strategies.

What we will learn:

- Local search methods
- Simulated annealing
- Genetic algorithms

Motivating Example: Subset regression

Suppose $x_1, \ldots, x_p$ are predictors that can be used to construct a linear model for Y. Each candidate model is

$$Y = \sum_{j=1}^{s} \beta_{i_j} x_{i_j} + \varepsilon,$$

where $1 \le i_1 < i_2 < \ldots < i_s \le p$, $s \ge 0$. The Akaike information criterion (AIC) for the candidate model is

$$\mathsf{AIC} = n \log \frac{\mathsf{RSS}}{n} + 2s$$

where n is the sample size and RSS is the residual sum of squares of model. The best model is the one that minimizes AIC.
To parameterize the model, set $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_p)$, such that $\theta_i = 1$ if x_i is included as a predictor, and 0 otherwise. The set $\boldsymbol{\Theta}$ of all possible $\boldsymbol{\theta}$ has 2^p values.

Local search

First, define a neighborhood $\mathcal{N}(\boldsymbol{\theta})$ for each $\boldsymbol{\theta}$, so that it

- contains candidate solutions that are "near" $\boldsymbol{\theta}$, and
- reduces the number of changes to the current $\boldsymbol{\theta}$.

For example, if $\boldsymbol{\Theta} = \{\boldsymbol{\theta} = (\theta_1, \ldots, \theta_p) : \text{each } \theta_i = 0, 1\}$, one may define $\mathcal{N}(\boldsymbol{\theta})$ to be the set of $\boldsymbol{\theta}'$ which are different from $\boldsymbol{\theta}$ in at most one coordinate.

After neighborhoods are defined, at iteration t, choose $\boldsymbol{\theta}_{t+1}$ from $\mathcal{N}(\boldsymbol{\theta}_t)$ according to a certain rule. For example,

- steepest ascent: $\boldsymbol{\theta}_{t+1} = \arg\max_{\boldsymbol{\theta} \in \mathcal{N}(\boldsymbol{\theta}_t)} f(\boldsymbol{\theta})$;
- ascent algorithm: $\boldsymbol{\theta}_{t+1} \in \mathcal{N}(\boldsymbol{\theta}_t)$ uphill from $\boldsymbol{\theta}_t$, i.e.,

$$f(\boldsymbol{\theta}_{t+1}) \geq f(\boldsymbol{\theta}_t).$$

To avoid trapping into a local maximum, two often used variants are

- random-starts local search: repeatedly run an ascent algorithm to termination from a large number of randomly chosen starting points.
- steepest ascent/mildest descent: set to the least unfavorable $\boldsymbol{\theta}_{t+1} \in \mathcal{N}(\boldsymbol{\theta}_t)$;
  - if $\boldsymbol{\theta}_t$ is a local maximum $\boldsymbol{\theta}_{t+1}$ is the one with least decrease;
  - otherwise, $\boldsymbol{\theta}_{t+1}$ is the one with the largest increase.

To select a linear model to minimize AIC (or maximize $-$AIC),

- randomly select a set of predictors
- at iteration t, decrease the AIC by adding a predictor to the set of selected predictors or deleting a predictor that has been selected; continue the iterations until the AIC can not be decreased;
- repeat Steps 1 and 2 many times, and choose the set of predictors at termination that has the lowest AIC.

A salesman must visit each of p cities exactly once and return to his starting city, using the shortest total travel distance.
A candidate solution is

$$\boldsymbol{\theta} = (i_1, i_2, \dots, i_p, i_1)$$

where $i_1, \dots, i_p$ is a permutation of 1, …, p. The objective function to be minimized is

$$f(\boldsymbol{\theta}) = d(i_1, i_2) + d(i_2, i_3) + \cdots + d(i_{p-1}, i_p) + d(i_p, i_1).$$

There are $(p-1)!/2$ all possible routes, since the point of origin and direction of travel are arbitrary. For a traveling plan to visit 20 cities, that amounts to more than 6×10^{16} possible routes.

One could define $\mathcal{N}(\boldsymbol{\theta})$ as the set of sequences that only differ from $\boldsymbol{\theta}$ at two entries. In other words, each sequence in $\mathcal{N}(\boldsymbol{\theta})$ is obtained by exchanging two entries of $\boldsymbol{\theta}$.

For example, if

$$\boldsymbol{\theta} = (1, 2, 5, 4, 6, 3, 1)$$

then

$$(1, 2, 3, 4, 6, 5, 1)$$

is a neighbor of $\boldsymbol{\theta}$, because it only has 3 and 5 exchanged; while

$$(1, 2, 4, 3, 6, 5, 1)$$

is not a neighbor of $\boldsymbol{\theta}$, because it has 5, 4, and 3 rotated.

Simulated annealing

In most cases, the simulated annealing algorithm can be thought of as a randomized local search algorithm. It uses a "temperature" parameter to control the randomness of the search. The algorithm starts with a high temperature and cools down gradually so that a global optimum may be reached. This is analogous to the annealing of metal or glass.

In the following description, a global *minimum* of f is being searched. The algorithm is run in stages, such that for the iterations within a stage, the temperature is a constant τ_j.

Suppose iteration t belongs to stage j.

1. Sample a candidate $\boldsymbol{\theta}^* \in \mathcal{N}(\boldsymbol{\theta})$ according to a *proposal density* $g_t(\boldsymbol{\theta} \mid \boldsymbol{\theta}_t)$; for different t, the proposal density g_t can be different.
2. Let $\Delta = f(\boldsymbol{\theta}^*) - f(\boldsymbol{\theta}_t)$.
   - a) If $\Delta \le 0$, then set $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}^*$.
   - b) If $\Delta > 0$, then set $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}^*$ with probability $e^{-\Delta/\tau_j}$, and $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t$ otherwise. This can be done as follows. Sample $U \sim$ Unif$(0, 1)$. Then

$$\boldsymbol{\theta}_{t+1} = \begin{cases} \boldsymbol{\theta}^* & \text{if} \quad U \le e^{-\Delta/\tau_j} \\ \boldsymbol{\theta}_t & \text{otherwise.} \end{cases}$$

3. Repeat steps 1 and 2 a total of m_j times.
4. Update $\tau_{j+1} = \alpha(\tau_j)$, $m_{j+1} = \beta(m_j)$ and move to stage $j + 1$, where α and β are two deterministic functions that govern how to cool down the temperature and how long to stay at a given temperature.

Suppose I is an image consisting of "pixels" $I(i,j)$, $i,j=1,\ldots,N$. The image is corrupted by noise $Z(i,j)$ and only $J=I+Z$ is observed. To reconstruct I, one way is to minimize a function

$$f(I)=\sum_{i,j}|J(i,j)-I(i,j)|^2+K(I),$$

where $K(I)$ is a function that has large values if I has many "irregularities". The idea is that, as long as the noise is not too strong, the real image should be similar to J; on the other hand, irregularities observed in J are likely to be due to noise and should be reduced.
One way to use the simulated annealing to minimize $f(I)$ is as follows. In each iteration, only one pixel of I can be updated. That means two I and I' are neighbors only when they are different at just one pixel. Then at each iteration, choose a pixel $I(i,j)$ and select a candidate value for it. Update $I(i,j)$ according to the rule in step 2 and move to the next iteration.

Some guidelines:

- R function

  `optim`

  can implement some simple versions of simulated annealing;
- the temperature τ_j should slowly decrease to 0;
- the number m_j of iterations at each temperature τ_j should be large and increasing in j;
- reheating strategies that allow sporadic, systematic, or interactive temperature increases to prevent getting stuck in a local minimum at low temperatures can be effective.

Genetic algorithms

First, each $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ is a string of alphabets:

$$\boldsymbol{\theta} = (\theta_1, \dots, \theta_C),$$

where each θ_i is a symbol from a finite set, such as $\{0, 1\}$, $\{0, 1, 2\}$, $\{$'a', 'b', ...$\}$.

Genetic algorithms regard the maximization of $f(\boldsymbol{\theta})$ over $\boldsymbol{\Theta}$ as a process of natural selection, with $f(\boldsymbol{\theta})$ being a measure of fitness and $\boldsymbol{\theta}$ the genetic code, or "chromosome". It assumes that fitness is a result of some "good" pieces of $\boldsymbol{\theta}$ and by inheriting these pieces plus some mutations fitness is enhanced.

In a genetic algorithm, at each iteration t, at least two candidate solutions $\boldsymbol{\theta}_{t,1}$, ..., $\boldsymbol{\theta}_{t,P}$ have to be tracked. Each iteration consists of several steps.

1. **Selection.** Randomly select from $\boldsymbol{\theta}_{t,1}, \ldots, \boldsymbol{\theta}_{t,P}$ to form a set of pairs. The selection should be based on a *fitness function* $\phi(\boldsymbol{\theta})$. In general, larger values of $f(\boldsymbol{\theta})$ result in larger values of $\phi(\boldsymbol{\theta})$, and higher chance for $\boldsymbol{\theta}$ to be selected.
   There are many selection mechanisms:
   - a) select one parent $\boldsymbol{\theta}$ with probability proportional to $\phi(\boldsymbol{\theta})$ and the other parent completely at random;
   - b) select each parent independently with probability proportional to $\phi(\boldsymbol{\theta})$;

   ϕ must be selected carefully: using $\phi(\boldsymbol{\theta}_{t,i}) = f(\boldsymbol{\theta}_{t,i})$ may result in rapid convergence into a local maximum. A common choice is

$$\phi(\boldsymbol{\theta}_{t,i}) = \frac{2r_i}{P(P+1)}$$

   where r_i is the *rank* of $f(\boldsymbol{\theta}_{t,i})$ in $f(\boldsymbol{\theta}_{t,1}), \ldots, f(\boldsymbol{\theta}_{t,P})$.

2. **Breeding.** For each pair, $(\boldsymbol{\theta}_a, \boldsymbol{\theta}_b)$, generate one or more "offspring" $\boldsymbol{\theta}' = c(\boldsymbol{\theta}_a, \boldsymbol{\theta}_b) \in \boldsymbol{\Theta}$, where c is a random operator.
   Typically, c is a "crossover". If

$$\boldsymbol{\theta}_a = (\theta_{a1}, \dots, \theta_{aC}),$$
$$\boldsymbol{\theta}_b = (\theta_{b1}, \dots, \theta_{bC}),$$

   then a crossover works as follows,

   a) randomly choose a position $1 \le d \le C$; and
   b) combine $(\theta_{a1}, \dots, \theta_{ad})$ and $(\theta_{b,d+1}, \dots, \theta_{bC})$ to form

$$\boldsymbol{\theta}' = (\theta_{a1}, \dots, \theta_{ad}, \theta_{b,d+1}, \dots, \theta_{bC}).$$

   If necessary, also take

$$\boldsymbol{\theta}'' = (\theta_{a,d+1}, \dots, \theta_{aC}, \theta_{b1}, \dots, \theta_{bd})$$

   to be another offspring.

3. **Mutation.** Make some random modifications to each offspring. Typically, if an offspring chromosome is

$$\boldsymbol{\theta} = (\theta_1, \ldots, \theta_C)$$

then for $i = 1, \ldots, C$, with a small probability $0 < p < 1$ (mutation rate), change θ_i to a different value.

The offspring produced at iteration t are taken as the candidate solutions for iteration $t + 1$.

Initialization.

Usually the first generation consists of completely random individuals.

- Large values of the size of a generation, P, are preferred. For binary encoding of $\boldsymbol{\theta}$, one suggestion is to have $C \leq P \leq 2C$, where C is the chromosome length. In most real applications, population sizes have ranged between 10 and 200.
- Mutation rates are typically very low, in the neighborhood of 1%.

Termination.

A genetic algorithm is usually terminated after a maximum number of iterations. Alternatively, the algorithm can be stopped once the genetic diversity in the current generation is sufficiently low.

In many problems, like the traveling salesman problem, it is natural to write $\boldsymbol{\theta}$ as a permutation of $1, 2, \ldots, p$.

- Standard crossover usually produces invalid sequences

For example, if

$$\boldsymbol{\theta}_a = \{7, 5, 2, 6, 3, 1, 9, 4, 8\}$$
$$\boldsymbol{\theta}_b = \{9, 1, 2, 3, 8, 6, 7, 5, 4\}$$

and if the crossover point is between 2nd and 3rd positions, then it produces

$$\{7, 5, 2, 3, 8, 6, 7, 5, 4\}$$

an invalid traveling route.

A remedy is order crossover: pick a number of positions from one parent and get the values at those positions, say $i_1, \ldots, i_s$. Next identify those values from the 2nd parent. Suppose the set of values are found at $j_1 < j_2 < \ldots < j_s$. Put i_1 at j_1, i_2 at j_2, ..., i_s at j_s while keeping the values of the 2nd on other locations unchanged.
Example: Consider parents (752631948) and (912386754) and random positions $(4, 6, 7)$. The offsprings are (612389754) and (352671948).
The drawback of the operation is that it destroys some important structures of the parent routes, in particular, the links.

Edge-recombination crossover uses a different idea. It generates an offspring whose edges belong to those of the parent routes. By edge it means a link into or out of a city in a route. For example, the above two parents have the following links, the order of numbers in each link is unimportant.

$$(1,2),(1,3),(1,9),(2,3),(2,5),(2,6),(3,6),(3,8),$$
$$(4,5),(4,8),(4,9),(5,7),(6,7),(6,8),(7,8)$$

First, choose one of the initial cities of the parents as the initial city of the offspring. At k-th step, if $i_1,\ldots,i_k$ have been chosen as the first k cities on the route, then choose among cities that are linked to i_k *and* are different from $i_1,\ldots,i_{k-1}$ as the $(k+1)$st city of the offspring.

EM and MM Algorithms

EM Optimizations

What it is for

- inference when there is missing information
  - incomplete observations
  - auxiliary parameters that are unobserved but can facilitate inference on the main parameter.
  - Some important applications of the EM algorithm are in problems where one has what we might call pseudo missing data. We never had any chance of obtaining such data, but we could pretend they are missing and use EM algorithm to facilitate the computation of MLEs.

Why it is important

- widely useful
- simple to implement
- numerically stable

Suppose a population consists of several sub-populations, each following a distribution $p(x \,|\, \theta_k)$ and having population fraction q_k. Typically,

$$\boldsymbol{q} = (q_1, \ldots, q_K), \quad \boldsymbol{\theta} = (\theta_1, \ldots, \theta_K)$$

are unknown, and the goal is to estimate them.

Let $x_1, \ldots, x_n$ be a sample. If for each x_i we know it comes from the z_i^{th} sub-population, then, letting

$$\boldsymbol{x} = (x_1, \ldots, x_n), \quad \boldsymbol{z} = (z_1, \ldots, z_n),$$

the likelihood function of $(\boldsymbol{q}, \boldsymbol{\theta})$ is

$$L(\boldsymbol{q}, \boldsymbol{\theta} \,|\, \boldsymbol{x}, \boldsymbol{z}) = \prod_{i=1}^{n} [q_{z_i} \times p(x_i \,|\, \theta_{z_i})].$$

However, in many cases, $\boldsymbol{z}$ is unobservable:

- cluster analysis
- the “sub-populations” may be constructs that facilitate inference or decision making, so they are nonexistent in reality.

The likelihood function then becomes

$$L(\boldsymbol{q}, \boldsymbol{\theta} \,|\, \boldsymbol{x}) = \sum_{\boldsymbol{z}} L(\boldsymbol{q}, \boldsymbol{\theta} \,|\, \boldsymbol{x}, \boldsymbol{z}).$$

The question is how to get the MLE of $\boldsymbol{q}$ and $\boldsymbol{\theta}$ efficiently.
In this example, $\boldsymbol{y} = (\boldsymbol{x}, \boldsymbol{z})$ is the complete data. When only $\boldsymbol{x}$ is observable, $\boldsymbol{z}$ is called the missing data.

Suppose a population follows a distribution $p(y \mid \theta)$, where θ is the unknown parameter. Suppose a random sample is collected from $p(y \mid \theta)$. However, when an observation y_i is greater than a cut-off C, it is truncated and recorded as C, such as in clinical trials. In this case, we only know $y_i \geq C$ but not its exact value. On the other hand, if $y_i \leq C$, then it is fully observed. Therefore, the observed data is

$$x_1 = \min(y_1, C), \ \ldots, \ x_n = \min(y_n, C).$$

In this case, $\boldsymbol{y} = (y_1, \ldots, y_n)$ is the complete data and $\boldsymbol{x} = (x_1, \ldots, x_n)$. The question is how to estimate θ based on $\boldsymbol{x}$.

Suppose $Z_1, \ldots, Z_n$ and $X_1, \ldots, X_n$ are time series related by

$$X_k = f(Z_k \,|\, \theta),$$

where the transformation f is parametrized by θ. Z_k may include not only variables of interest, but also some “noise” that interact with the variables. Suppose the joint distribution of $Z_1, \ldots, Z_n$ has the parametric form $h(z_1, \ldots, z_n \,|\, \gamma)$, however, the values of θ and γ are both unknown. If $X_1, \ldots, X_n$ are observed but $Z_1, \ldots, Z_n$ are only partially observed or completely hidden, how to estimate θ and γ?

The Hidden Markov Model (HMM) is a typical case. Let $S_1, \ldots, S_n$ be a discrete Markov chain with a finite number of states. Conditional on $S_1, \ldots, S_n$, the observations $X_1, \ldots, X_n$ are independent, such that, given $S_k = s$, $X_k \sim N(\mu_s, \sigma_s^2)$.

$$\begin{tikzcd}
S_1 \arrow[r] \arrow[d] & S_2 \arrow[r] \arrow[d] & S_3 \arrow[r] \arrow[d] & S_4 \arrow[r] \arrow[d] & \cdots \\
X_1 & X_2 & X_3 & X_4 &
\end{tikzcd}$$

In this case,

$$Z_k = (S_k, W_k),$$

where $W_1, \ldots, W_n \sim N(0,1)$ are independent of $Z_1, \ldots, Z_n$, such that $X_k = \mu_{S_k} + \sigma_{S_k} W_k$. Typically, μ_k, σ_k and the transition probabilities of S_k are unknown and only X_k are observed.

Positron emission tomography (PET) is a tool to study the metabolic activity in organs. To generate a PET scan, some radioactive material is administered into the organ. Then the emissions of the material are recorded using a PET scanner, which consists of an array of detectors surrounding the patient's body. The region of interest is divided into "voxels". The number of photons Y_{ij} coming from voxel i and received by detector j is assumed to follow a Poisson distribution

$$Y_{ij} \sim \text{Poisson}(\theta_i a_{ij})$$

where the coefficient a_{ij} can be determined accurately beforehand, and θ_i is the emission density of the radioactive material in voxel i, which is used to quantify the metabolic activity therein.

If for each pair of voxel and detector, we can observe Y_{ij}, then θ_j's can be estimated by MLE. However, in reality, each detector j receives photons from all the voxels, and hence only the sum

$$X_j = \sum_i Y_{ij}$$

is observable. The goal is to use $\boldsymbol{X} = (X_j)$ instead of $\boldsymbol{Y} = (Y_{ij})$ to estimate $\boldsymbol{\theta} = (\theta_j)$.

Suppose the complete data

$$\boldsymbol{Y} \sim p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta})$$

while the observed data is $\boldsymbol{X} = M(\boldsymbol{Y})$, where M is a many-to-one mapping. Then

$$\underbrace{p_{\boldsymbol{X}}(\boldsymbol{x} \,|\, \boldsymbol{\theta})}_{L(\boldsymbol{\theta} \,|\, \boldsymbol{x})} = \int_{M(\boldsymbol{y})=\boldsymbol{x}} \underbrace{p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta})}_{L(\boldsymbol{\theta} \,|\, \boldsymbol{y})} \, \mathrm{d}\boldsymbol{y}.$$

If only $\boldsymbol{X} = \boldsymbol{x}$ is observed, then the MLE $\widehat{\theta}$ satisfies

$$L'(\widehat{\theta} \,|\, \boldsymbol{x}) = 0$$

i.e.,

$$\int_{M(\boldsymbol{y})=\boldsymbol{x}} L'(\widehat{\theta} \,|\, \boldsymbol{y}) \, \mathrm{d}\boldsymbol{y} = 0.$$

Write log-likelihood function $l(\boldsymbol{\theta} \mid \boldsymbol{y}) = \ln L(\boldsymbol{\theta} \mid \boldsymbol{y})$, so that

$$[L(\boldsymbol{\theta} \mid \boldsymbol{y})]' = [e^{l(\boldsymbol{\theta} \mid \boldsymbol{y})}]' = [l(\boldsymbol{\theta} \mid \boldsymbol{y})]' e^{l(\boldsymbol{\theta} \mid \boldsymbol{y})} = [l(\boldsymbol{\theta} \mid \boldsymbol{y})]' L(\boldsymbol{\theta} \mid \boldsymbol{y}).$$

Therefore,

$$\int_{M(\boldsymbol{y})=\boldsymbol{x}} L'(\widehat{\theta} \mid \boldsymbol{y}) \,\mathrm{d}\boldsymbol{y} = 0$$

is the same as

$$\int_{M(\boldsymbol{y})=\boldsymbol{x}} [l(\widehat{\theta} \mid \boldsymbol{y})]' L(\widehat{\theta} \mid \boldsymbol{y}) \ \mathrm{d}\boldsymbol{y} = 0.$$

In principle, Newton's method can be used to find a solution. However, since $\widehat{\theta}$ appears in two places in the integration, the implementation is often messy and because of that, numerically unstable.

We may try to "decouple" the two $\boldsymbol{\theta}$'s in the integration to get simpler iterations. An iterative procedure could go as follows. At step t, if we have $\boldsymbol{\theta}_t$, then find $\boldsymbol{\theta}_{t+1}$ to solve

$$\int_{M(\boldsymbol{y})=\boldsymbol{x}} [l(\boldsymbol{\theta}_{t+1} \mid \boldsymbol{y})]' L(\boldsymbol{\theta}_t \mid \boldsymbol{y}) \, \mathrm{d}\boldsymbol{y} = 0,$$

or, to optimize the function

$$\int_{M(\boldsymbol{y})=\boldsymbol{x}} l(\boldsymbol{\theta} \mid \boldsymbol{y}) L(\boldsymbol{\theta}_t \mid \boldsymbol{y}) \, \mathrm{d}\boldsymbol{y}.$$

This is a function in $\boldsymbol{\theta}$. Since $\boldsymbol{\theta}_t$ is different at each step, the function is different. If $\boldsymbol{\theta}_t$ converge, then the limit is a solution to the original MLE.

The explicit numerical integration in the above function is inconvenient. To replace it, note that, since $\boldsymbol{\theta}_t$ is fixed, the above way to find $\boldsymbol{\theta}_{t+1}$ is equivalent to optimizing

$$\int_{M(\boldsymbol{y})=\boldsymbol{x}} l(\boldsymbol{\theta} \,|\, \boldsymbol{y}) \frac{L(\boldsymbol{\theta}_t \,|\, \boldsymbol{y})}{L(\boldsymbol{\theta}_t \,|\, \boldsymbol{x})} \, \mathrm{d}\boldsymbol{y}.$$

For any $\boldsymbol{\theta}_t$,

$$\frac{L(\boldsymbol{\theta}_t \,|\, \boldsymbol{y})}{L(\boldsymbol{\theta}_t \,|\, \boldsymbol{x})} = \frac{p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta}_t)}{p_{\boldsymbol{X}}(\boldsymbol{x} \,|\, \boldsymbol{\theta}_t)} = p_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{y} \,|\, \boldsymbol{x}, \boldsymbol{\theta}_t),$$

the conditional density of $\boldsymbol{Y}$ given $\boldsymbol{X} = \boldsymbol{x}$ under $p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta}_t)$. Then the integral is simply $\mathbb{E}\left[l(\boldsymbol{\theta} \,|\, \boldsymbol{Y}) \,|\, \boldsymbol{x}, \boldsymbol{\theta}_t\right]$. In turns out that in order to find suitable $\boldsymbol{\theta}_{t+1}$, we need to maximize this function.

EM (Expectation-Maximization) algorithm.

Define

$$Q(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}') = \mathbb{E}\left[l(\boldsymbol{\theta} \mid \boldsymbol{Y}) \,|\, \boldsymbol{x}, \boldsymbol{\theta}' \right] = \mathbb{E}\left[\ln L(\boldsymbol{\theta} \mid \boldsymbol{Y}) \,|\, \boldsymbol{x}, \boldsymbol{\theta}' \right].$$

- Initialize $\boldsymbol{\theta}_0$.
- At iteration $t = 1, 2, \ldots$, with $\boldsymbol{\theta}_t$ given, set $\boldsymbol{\theta}_{t+1}$ to maximize $Q(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}_t)$.

The "E step" of the algorithm refers to the computation of $Q(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}_t)$, and the "M step" refers to the maximization of $Q(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}_t)$. Stopping criteria are usually based upon $|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t|$ or $|Q(\boldsymbol{\theta}_{t+1} \,|\, \boldsymbol{\theta}_t) - Q(\boldsymbol{\theta}_t \,|\, \boldsymbol{\theta}_t)|$.

If $\boldsymbol{Y} = (\boldsymbol{X}, \boldsymbol{Z})$, where $\boldsymbol{Z}$ is the missing data, then, given $\boldsymbol{X} = \boldsymbol{x}$, the only unknown part of $\boldsymbol{Y}$ is $\boldsymbol{Z}$, so

$$
\begin{aligned}
Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}') &= \mathbb{E}\left[\ln L(\boldsymbol{\theta} \mid \boldsymbol{Y}) \mid \boldsymbol{x}, \boldsymbol{\theta}'\right] \\
&= \mathbb{E}\left[\ln L(\boldsymbol{\theta} \mid \boldsymbol{x}, \boldsymbol{Z}) \mid \boldsymbol{x}, \boldsymbol{\theta}'\right] \\
&= \begin{cases} \displaystyle\sum_{\boldsymbol{z}} p(\boldsymbol{z} \mid \boldsymbol{x}, \boldsymbol{\theta}') \ln p(\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}), & \text{if } \boldsymbol{Z} \text{ is discrete} \\ \displaystyle\int p(\boldsymbol{z} \mid \boldsymbol{x}, \boldsymbol{\theta}') \ln p(\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{z}, & \text{if } \boldsymbol{Z} \text{ is continuous} \end{cases}
\end{aligned}
$$

Gaussian mixture clustering

Suppose a population is a mixture of Gaussian distributions $N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ with population fractions q_k, $k = 1, \dots, K$. The goal is to estimate $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$, q_k.

Let $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n \in \mathbb{R}^d$ be an iid sample from the population. Let z_i be the index of the Gaussian distribution from which $\boldsymbol{x}_i$ is sampled. Suppose that

$$\boldsymbol{z} = (z_1, \dots, z_n)$$

cannot be observed. The observed data thus is

$$\boldsymbol{x} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n).$$

Suppose at iteration t of the EM algorithm, one has obtained

$$\boldsymbol{\theta}_t = (\boldsymbol{\mu}_{t1}, \dots, \boldsymbol{\mu}_{tK}, \boldsymbol{\Sigma}_{t1}, \dots \boldsymbol{\Sigma}_{tK}, q_{t1}, \dots, q_{tK}).$$

Then, since $(\boldsymbol{x}_i, z_i)$ are independent, and z_i are discrete, for any candidate

$$\boldsymbol{\theta} = (\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_1, \dots, \boldsymbol{\Sigma}_K, q_1, \dots, q_K),$$

one has

$$\begin{aligned} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_t) &= \sum_{\boldsymbol{z}} p(\boldsymbol{z} \mid \boldsymbol{x}, \boldsymbol{\theta}_t) \ln p(\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}) \\ &= \sum_{i=1}^{n} \sum_{k=1}^{K} \underbrace{p(z_i = k \mid \boldsymbol{x}_i, \boldsymbol{\theta}_t)}_{w_{ik}} \ln p(z_i = k, \boldsymbol{x}_i \mid \boldsymbol{\theta}). \end{aligned}$$

To maximize $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_t)$, note that w_{ik} has nothing to do with $\boldsymbol{\theta}$ so it should be computed in the first place. By Bayes rule,

$$w_{ik} = \frac{p(z_i = k, \boldsymbol{x}_i \mid \boldsymbol{\theta}_t)}{p(\boldsymbol{x}_i \mid \boldsymbol{\theta}_t)} = \frac{p(z_i = k, \boldsymbol{x}_i \mid \boldsymbol{\theta}_t)}{\sum_{s=1}^{K} p(z_i = s, \boldsymbol{x}_i \mid \boldsymbol{\theta}_t)},$$

with

$$\begin{aligned} p(z_i = k, \boldsymbol{x}_i \mid \boldsymbol{\theta}_t) &= p(z_i = k \mid \boldsymbol{\theta}_t) p(\boldsymbol{x}_i \mid z_i = k, \boldsymbol{\theta}_t) \\ &= q_{tk} \phi(\boldsymbol{x}_i \mid \boldsymbol{\mu}_{tk}, \boldsymbol{\Sigma}_{tk}), \end{aligned} \tag{1}$$

where $\phi(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the density of $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ at $\boldsymbol{x}$. Once w_{ik}'s are computed,

$$Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_t) = \sum_{k=1}^{K} \sum_{i=1}^{n} w_{ik} \ln p(z_i = k, \boldsymbol{x}_i \mid \boldsymbol{\theta}),$$

which looks a lot simpler.

As in (1), $p(z_i = k, \boldsymbol{x}_i \mid \boldsymbol{\theta}) = q_k n(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$. Since

$$n(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{(2\pi)^{-d/2}}{\sqrt{|\boldsymbol{\Sigma}_k|}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)'\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)\right\},$$

then

$$\begin{aligned} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_t) &= \sum_{k=1}^{K}\sum_{i=1}^{n} w_{ik} \ln[(2\pi)^{-d/2} q_k] - \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{n} w_{ik} \ln|\boldsymbol{\Sigma}_k| \\ &\quad - \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{n} w_{ik}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)'\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_k) \\ &= I_1 - \frac{I_2}{2} - \frac{I_3}{2}. \end{aligned}$$

From last page,

$$I_1 = \sum_{k=1}^{K}\sum_{i=1}^{n} w_{ik} \ln[(2\pi)^{-d/2} q_k], \quad I_2 = \sum_{k=1}^{K}\sum_{i=1}^{n} w_{ik} \ln|\boldsymbol{\Sigma}_k|$$

$$I_3 = \sum_{k=1}^{K}\sum_{i=1}^{n} w_{ik}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)'\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)$$

First, only I_3 contains $\boldsymbol{\mu}_k$, which is a quadratic form. To *minimize* it, observe that I_3 is the sum of K quadratic forms, each involving a single $\boldsymbol{\mu}_k$

$$I_{3k} = \sum_{i=1}^{n} w_{ik}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)'\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_k).$$

We only need to find $\boldsymbol{\mu}_k$ to minimize each I_{3k}. From the property of sample mean, $\boldsymbol{\mu}_k$ must be the mean of a weighted sample $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, each $\boldsymbol{x}_i$ having weight w_{ik}. So

$$\boldsymbol{\mu}_k = \frac{\sum_{i=1}^{n} w_{ik}\boldsymbol{x}_i}{\sum_{i=1}^{n} w_{ik}}.$$

Next, only I_2 and I_3 contain $\boldsymbol{\Sigma}_k$, and $I_2 + I_3$ is the sum of K terms of the following form, each involving a single $\boldsymbol{\Sigma}_k$,

$$S_k = \sum_{i=1}^{n} w_{ik} \ln |\boldsymbol{\Sigma}_k| + \sum_{i=1}^{n} w_{ik}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)' \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)$$

and so we only need to find $\boldsymbol{\Sigma}_k$ to minimize each S_k. Now that $\boldsymbol{\mu}_k$ is equal to the weighted mean of $\boldsymbol{x}_i$, to *minimize* S_k, $\boldsymbol{\Sigma}_k$ must be the sample variance of the weighted sample $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$. So

$$\boldsymbol{\Sigma}_k = \frac{\sum_{i=1}^{n} w_{ik}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)'}{\sum_{i=1}^{n} w_{ik}}.$$

Finally, only I_1 contains q_k. Since

$$\begin{aligned} I_1 &= \sum_{k=1}^K \sum_{i=1}^n w_{ik} \ln[(2\pi)^{-d/2} q_k] \\ &= \sum_{k=1}^K \sum_{i=1}^n w_{ik} \ln[(2\pi)^{-d/2}] + \sum_{k=1}^K \sum_{i=1}^n w_{ik} \ln q_k \\ &= -(d/2) \ln(2\pi) \sum_{k=1}^K \sum_{i=1}^n w_{ik} + \sum_{k=1}^K \left(\sum_{i=1}^n w_{ik} \right) \ln q_k \end{aligned}$$

it suffices to minimize

$$\sum_{k=1}^K \underbrace{\left(\sum_{i=1}^n w_{ik} \right)}_{W_k} \ln q_k$$

Note $q_1, \ldots, q_K$ must satisfy $q_1 + \cdots + q_K = 1$. Therefore, the maximization can be obtained by finding a solution for $\mathcal{L}'(q_1, \ldots, q_K) = 0$, i.e.

$$\frac{\partial \mathcal{L}(q_1, \ldots, q_K)}{\partial q_k} = 0$$

where

$$\mathcal{L}(q_1, \ldots, q_K) = \sum_{k=1}^{K} W_k \ln q_k - \lambda \left\{ \sum_{k=1}^{K} q_k - 1 \right\}$$

with λ a Lagrange multiplier. Then

$$q_k = \frac{W_k}{\sum_{k=1}^{K} W_k}.$$

Since for each i, $\sum_{k=1}^{K} w_{ik} = 1$,

$$q_k = \frac{1}{n} \sum_{i=1}^{n} w_{ik}.$$

The above steps consist a basic EM Gaussian clustering algorithm. It can be summarized as follows.

- Given

  $$\boldsymbol{\theta}_t = (\boldsymbol{\mu}_{t1}, \ldots, \boldsymbol{\mu}_{tn}, \boldsymbol{\Sigma}_{t1}, \ldots \boldsymbol{\Sigma}_{tK}, q_{t1}, \ldots, q_{tK}),$$

  for $k = 1, \ldots, K$, compute

  $$w_{ik} = p(z_i = k \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_t), \quad i = 1, \ldots, n$$

  then set $q_{t+1,k}$ equal to the mean of w_{1k}, . . . , w_{nk}, $\boldsymbol{\mu}_{t+1,k}$ and $\boldsymbol{\Sigma}_{t+1,k}$ equal to the weighted average and weighted covariance of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, with weights w_{1k}, . . . , w_{nk}, respectively.

For high dimensional data, there are many variants of the algorithm which put constraints on $\boldsymbol{\Sigma}_k$ to improve computational and estimation efficiency. The strongest constraint (and computationally the simplest) is

$$\boldsymbol{\Sigma}_1 = \cdots = \boldsymbol{\Sigma}_K = \sigma^2 \boldsymbol{I},$$

with σ^2 being the only parameter. Others include

1. $\boldsymbol{\Sigma}_k = \sigma_k^2 \boldsymbol{I}$, $k = 1, \ldots, K$, with σ_k being the parameters;
2. $\boldsymbol{\Sigma}_k$ are diagonal matrices that may be equal to or different from each other;
3. $\boldsymbol{\Sigma}_1 = \sigma_k^2 \boldsymbol{\Sigma}$, with the parameters being σ_k^2 and $\boldsymbol{\Sigma}$.

If Newton's method is used to directly maximize the log likelihood function $l(\boldsymbol{\theta} \mid \boldsymbol{x}) = \ln L(\boldsymbol{\theta} \mid \boldsymbol{x})$, then one would have to differentiate the function

$$l(\boldsymbol{\theta} \mid \boldsymbol{x}) = \sum_{i=1}^{n} l_i(\boldsymbol{\theta})$$

where, for every $i = 1, \ldots, n$,

$$\begin{aligned} l_i(\boldsymbol{\theta}) = \ln p(\boldsymbol{x}_i \mid \boldsymbol{\theta}) &= \ln \left\{ \sum_{z_i} p(\boldsymbol{x}_i, z_i \mid \boldsymbol{\theta}) \right\} \\ &= \ln \left\{ \sum_{k=1}^{K} p(\boldsymbol{x}_k \mid z_i = k, \boldsymbol{\theta}) p(z_i = k \mid \boldsymbol{\theta}) \right\} \\ &= \ln \left\{ \sum_{k=1}^{K} \frac{(2\pi)^{-\frac{d}{2}} q_k}{\sqrt{|\boldsymbol{\Sigma}_k|}} e^{-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)' \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)} \right\}. \end{aligned}$$

The iterations apparently are quite involved.

Computation

The R package

```
mclust
```

provides a large number of clustering methods, including EM based methods. Use

```
install.packages("mclust")
```

to install it on your computer if it is not yet installed. To use, first execute

```
library("mclust")
```

which loads all the functions in the package into an R session.

`mclustBIC`

computes the Bayes information criterion associated with different Gaussian mixture models, which are characterized by number of clusters and constraints on the covariance matrices.

`mclustModel`

estimate the parameters for the best model determined via `mclustBIC`.

`mclustModelNames`

lists the names of models used in the `mclust` package.

`mclust2Dplot`

plots 2-D data given parameters of a Gaussian mixture model for the data. Some MATLAB routines, such as `EM-GM.m` can perform some of the functions of the R package.

EM for Exponential Families

Many important parametric distributions belong to an exponential family, which has the form

$$p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta}) = c_1(\boldsymbol{y}) c_2(\boldsymbol{\theta}) \exp\left\{\boldsymbol{\theta}^T \boldsymbol{s}(\boldsymbol{y})\right\},$$

where $\boldsymbol{\theta}$ is a vector of parameters and $\boldsymbol{s}(\boldsymbol{y})$ a vector of sufficient statistics.

1. The normal densities $N_d(\boldsymbol{\theta}, \boldsymbol{\Sigma})$ with $\boldsymbol{\Sigma}$ being fixed consist an exponential family, because

$$\begin{aligned} p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta}) &= \frac{e^{-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\theta})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{y}-\boldsymbol{\theta})}}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \\ &= \underbrace{\frac{e^{-\frac{1}{2}\boldsymbol{y}^T\boldsymbol{y}}}{(2\pi)^{d/2}}}_{c_1(\boldsymbol{y})} \underbrace{\frac{e^{-\frac{1}{2}\boldsymbol{\theta}^T\boldsymbol{\theta}}}{|\boldsymbol{\Sigma}|^{1/2}}}_{c_2(\boldsymbol{\theta})} \exp\left\{\boldsymbol{\theta}^T \underbrace{(\boldsymbol{\Sigma}^{-1}\boldsymbol{y})}_{\boldsymbol{s}(\boldsymbol{y})}\right\}. \end{aligned}$$

1. Poisson distribution Poisson(θ), where $\theta > 0$, has distribution function

$$p(y \,|\, \theta) = \frac{e^{-\theta} y^{\theta}}{y!}, \quad y = 0, 1, \ldots$$

If we define

$$c_1(y) = \frac{1}{y!} \quad c_2(\theta) = e^{-\theta} \quad s(y) = \ln y,$$

then $p(y \,|\, \theta) = c_1(y) c_2(\theta) e^{\theta s(y)}$. Therefore, Poisson($\theta$) also form an exponential family.

1. Gamma distribution Gamma(a, r), where $a > 0$, $r > 0$, has density

$$p(y \mid a, r) = \begin{cases} \dfrac{e^{-y/r} y^{a-1}}{r^a \Gamma(a)} & y > 0 \\ 0 & y \leq 0. \end{cases}$$

Indeed, if we define $b = 1/r$, $\theta = (a, b)$, and

$$c_1(y) = \frac{1}{y} \quad c_2(\boldsymbol{\theta}) = c_2(a, b) = \frac{b^a}{\Gamma(\theta_1)} \quad \boldsymbol{s}(y) = (\ln y, -y)$$

Then

$$c_1(y) c_2(\boldsymbol{\theta}) \exp(\boldsymbol{\theta}^T \boldsymbol{s}(y)) = \frac{1}{y} \frac{1}{b^a \Gamma(a)} e^{a \ln y - by} = p(y \mid a, r).$$

1. Beta distribution Beta(a, b), where $a > 0$, $b > 0$, has density

$$p(y \mid a, b) = \begin{cases} \dfrac{y^{a-1}(1-y)^{b-1}}{B(a, b)}, & 0 < y < 1 \\ 0 & \text{otherwise} \end{cases}$$

If we let $\boldsymbol{\theta} = (a, b)$,

$$c_1(y) = \frac{1}{y(1-y)} \quad c_2(\boldsymbol{\theta}) = \frac{1}{B(a, b)} \quad \boldsymbol{s}(y) = (\ln y, \ln(1-y))$$

then

$$c_1(y)c_2(\boldsymbol{\theta})\exp(\boldsymbol{\theta}^T\boldsymbol{s}(y)) = \frac{1}{y(1-y)}\frac{1}{B(a, b)}e^{a\ln y + b\ln(1-y)} = p(y \mid a, b).$$

Suppose $\boldsymbol{x} = M(\boldsymbol{y})$ is observed and we want to estimate $\boldsymbol{\theta}$. The EM algorithm in this case is significantly simplified, with no explicit evaluation of $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}')$. It works as follows. Given $\boldsymbol{\theta}_t$, set $\boldsymbol{\theta}_{t+1}$ such that

$$\mathbb{E}\left[s(\boldsymbol{Y}) \mid \boldsymbol{\theta}_{t+1}\right] = \mathbb{E}\left[s(\boldsymbol{Y}) \mid \boldsymbol{x}, \boldsymbol{\theta}_t\right]. \tag{2}$$

To get the result, we start with the evaluation

$$\begin{aligned}
Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_t) &= \int [\ln p_{\boldsymbol{Y}}(\boldsymbol{y} \mid \boldsymbol{\theta})] p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}_t) \,\mathrm{d}\boldsymbol{y} \\
&= \int [\ln c_1(\boldsymbol{y}) + \ln c_2(\boldsymbol{\theta}) + \boldsymbol{\theta}^T \boldsymbol{s}(\boldsymbol{y})] p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}_t) \,\mathrm{d}\boldsymbol{y} \\
&= \int [\ln c_1(\boldsymbol{y})] p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}_t) \,\mathrm{d}\boldsymbol{y} \\
&\qquad + \int [\ln c_2(\boldsymbol{\theta})] p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}_t) \,\mathrm{d}\boldsymbol{y} + \int \boldsymbol{\theta}^T \boldsymbol{s}(\boldsymbol{y}) p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}_t) \,\mathrm{d}\boldsymbol{y} \\
&= k(\boldsymbol{\theta}_t) + \ln c_2(\boldsymbol{\theta}) + \boldsymbol{\theta}^T \mathbb{E}\left[s(\boldsymbol{Y}) \mid \boldsymbol{x}, \boldsymbol{\theta}_t\right].
\end{aligned}$$

To maximize $Q(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}_t)$ as a function of $\boldsymbol{\theta}$, it suffices to solve

$$(\ln c_2(\boldsymbol{\theta}))' + \mathbb{E}\left[s(\boldsymbol{Y}) \,|\, \boldsymbol{x}, \boldsymbol{\theta}_t\right] = 0,$$

or

$$\frac{c_2'(\boldsymbol{\theta})}{c_2(\boldsymbol{\theta})} + \mathbb{E}\left[s(\boldsymbol{Y}) \,|\, \boldsymbol{x}, \boldsymbol{\theta}_t\right] = 0.$$

The following equality is basic for exponential families. For *any* $\boldsymbol{\theta}$,

$$\frac{c_2'(\boldsymbol{\theta})}{c_2(\boldsymbol{\theta})} = -\mathbb{E}\left[\boldsymbol{s}(\boldsymbol{Y}) \,|\, \boldsymbol{\theta}\right] = -\int \boldsymbol{s}(\boldsymbol{y}) p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{y}. \tag{3}$$

Assuming the result is correct for now, it is seen that $\boldsymbol{\theta}_{t+1}$ should be set such that $\mathbb{E}\left[s(\boldsymbol{Y}) \,|\, \boldsymbol{\theta}_{t+1}\right] = \mathbb{E}\left[s(\boldsymbol{Y}) \,|\, \boldsymbol{x}, \boldsymbol{\theta}_t\right]$, i.e., as in (2).

To get (3), notice $\int p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta}) = 1$ for any $\boldsymbol{\theta}$, i.e.

$$c_2(\boldsymbol{\theta}) \int c_1(\boldsymbol{y}) \exp\{\boldsymbol{\theta}^T \boldsymbol{s}(\boldsymbol{y})\} \, \mathrm{d}\boldsymbol{y} = 1.$$

Then

$$\ln c_2(\boldsymbol{\theta}) = -\ln \left\{ \int c_1(\boldsymbol{y}) \exp\{\boldsymbol{\theta}^T \boldsymbol{s}(\boldsymbol{y})\} \, \mathrm{d}\boldsymbol{y} \right\}.$$

Differentiate both sides to get

$$\begin{aligned} \frac{c_2'(\boldsymbol{\theta})}{c_2(\boldsymbol{\theta})} &= -\frac{\int c_1(\boldsymbol{y}) \boldsymbol{s}(\boldsymbol{y}) \exp\{\boldsymbol{\theta}^T \boldsymbol{s}(\boldsymbol{y})\} \, \mathrm{d}\boldsymbol{y}}{\int c_1(\boldsymbol{y}) \exp\{\boldsymbol{\theta}^T \boldsymbol{s}(\boldsymbol{y})\} \, \mathrm{d}\boldsymbol{y}} \\ &= -c_2(\boldsymbol{\theta}) \int c_1(\boldsymbol{y}) \boldsymbol{s}(\boldsymbol{y}) \exp\{\boldsymbol{\theta}^T \boldsymbol{s}(\boldsymbol{y})\} \, \mathrm{d}\boldsymbol{y} \\ &= -\int \boldsymbol{s}(\boldsymbol{y}) p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{y}, \end{aligned}$$

as claimed.

Convergence of EM

We first show that each step of the algorithm increases the observed-data log likelihood $l(\boldsymbol{\theta} \mid \boldsymbol{x}) = \ln p_{\boldsymbol{X}}(\boldsymbol{x} \mid \boldsymbol{\theta})$, that is

$$l(\boldsymbol{\theta}_{t+1} \mid \boldsymbol{x}) \geq l(\boldsymbol{\theta}_t \mid \boldsymbol{x}).$$

Once this is done, one can say $l(\boldsymbol{\theta}_t \mid \boldsymbol{x})$ always climbs up to a local maximum of $l(\boldsymbol{\theta} \mid \boldsymbol{x})$.

- In general, there is no guarantee that the limit is the (global) maximum.
- EM has slower convergence than the Newton's method, sometime substantially.
- Nevertheless, the ease of implementation and the stable ascent of EM are often very attractive.

Convergence of EM

By $\boldsymbol{X} = M(\boldsymbol{Y})$, $p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta}) = p(\boldsymbol{x}, \boldsymbol{y} \,|\, \boldsymbol{\theta}) = p_{\boldsymbol{X}}(\boldsymbol{x} \,|\, \boldsymbol{\theta}) p_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{y} \,|\, \boldsymbol{x}, \boldsymbol{\theta})$, so

$$p_{\boldsymbol{X}}(\boldsymbol{x} \,|\, \boldsymbol{\theta}) = \frac{p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta})}{p_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{y} \,|\, \boldsymbol{x}, \boldsymbol{\theta})}.$$

Because $\boldsymbol{x}$ is fixed, $p_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{y} \,|\, \boldsymbol{x}, \boldsymbol{\theta})$ only depends on $\boldsymbol{\theta}$. To simplify notation, write $f(\boldsymbol{y} \,|\, \boldsymbol{\theta}) = p_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{y} \,|\, \boldsymbol{x}, \boldsymbol{\theta})$. Then

$$\ln p_{\boldsymbol{X}}(\boldsymbol{x} \,|\, \boldsymbol{\theta}) = \ln p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta}) - \ln f(\boldsymbol{y} \,|\, \boldsymbol{\theta}).$$

Convergence of EM

Integrate both sides with respect to the density $f(\boldsymbol{y} \,|\, \boldsymbol{\theta}_t)$. From

$$\begin{aligned}
&\int [\ln p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta})] \, f(\boldsymbol{y} \,|\, \boldsymbol{\theta}_t) \, \mathrm{d}\boldsymbol{y} \\
&= \int [\ln p_{\boldsymbol{Y}}(\boldsymbol{y} \,|\, \boldsymbol{\theta})] \, p_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{y} \,|\, \boldsymbol{x}, \boldsymbol{\theta}_t) \, \mathrm{d}\boldsymbol{y} = Q(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}_t),
\end{aligned}$$

It follows

$$\ln p_{\boldsymbol{X}}(\boldsymbol{x} \,|\, \boldsymbol{\theta}_t) = Q(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}_t) - H(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}_t), \tag{4}$$

where

$$H(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}_t) = \int [\ln f(\boldsymbol{y} \,|\, \boldsymbol{\theta})] f(\boldsymbol{y} \,|\, \boldsymbol{\theta}_t) \, \mathrm{d}\boldsymbol{y}.$$

Convergence of EM

To show $\ln p_{\boldsymbol{X}}(\boldsymbol{x} \,|\, \boldsymbol{\theta}_{t+1}) \geq \ln p_{\boldsymbol{X}}(\boldsymbol{x} \,|\, \boldsymbol{\theta}_t)$, by (4), their difference equals

$$[Q(\boldsymbol{\theta}_{t+1} \,|\, \boldsymbol{\theta}_t) - Q(\boldsymbol{\theta}_t \,|\, \boldsymbol{\theta}_t)] - [H(\boldsymbol{\theta}_{t+1} \,|\, \boldsymbol{\theta}_t) - H(\boldsymbol{\theta}_t \,|\, \boldsymbol{\theta}_t)].$$

By the M-step, $Q(\boldsymbol{\theta}_{t+1} \,|\, \boldsymbol{\theta}_t) \geq Q(\boldsymbol{\theta}_t \,|\, \boldsymbol{\theta}_t)$. On the other hand, by the Jensen's inequality,

$$\begin{aligned} H(\boldsymbol{\theta}_{t+1} \,|\, \boldsymbol{\theta}_t) - H(\boldsymbol{\theta}_t \,|\, \boldsymbol{\theta}_t) &= \int \ln \left[\frac{f(\boldsymbol{y} \,|\, \boldsymbol{\theta}_{t+1})}{f(\boldsymbol{y} \,|\, \boldsymbol{\theta}_t)} \right] f(\boldsymbol{y} \,|\, \boldsymbol{\theta}_t) \, \mathrm{d}\boldsymbol{y} \\ &\leq \ln \left[\int \frac{f(\boldsymbol{y} \,|\, \boldsymbol{\theta}_{t+1})}{f(\boldsymbol{y} \,|\, \boldsymbol{\theta}_t)} f(\boldsymbol{y} \,|\, \boldsymbol{\theta}_t) \, \mathrm{d}\boldsymbol{y} \right] = \ln \left[\int f(\boldsymbol{y} \,|\, \boldsymbol{\theta}_{t+1}) \, \mathrm{d}\boldsymbol{y} \right] = 0, \end{aligned}$$

thus showing the likelihood increases at each iteration.

Variants of EM

- E step is hard: Monte Carlo (MCEM)
- M step is hard: Instead of choosing $\boldsymbol{\theta}_{t+1}$ to maximize $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_t)$, one simply chooses any $\boldsymbol{\theta}_{t+1}$ for which

  $$Q(\boldsymbol{\theta}_{t+1} \mid \boldsymbol{\theta}_t) > Q(\boldsymbol{\theta}_t \mid \boldsymbol{\theta}_t),$$

  then the resulting algorithm is called generalized EM, or GEM.
  - In GEM, $l(\boldsymbol{\theta}_t \mid \boldsymbol{x})$ is still increasing.
  - An example: ECM algorithm, which replaces the M-step of the EM algorithm with several computationally simpler conditional maximization (CM) steps.

Variance Estimation

- Louis' method (1982, JRSSB)
- Supplemented EM (SEM) algorithm (Meng and Rubin, 1991, JASA)
- Oakes' method (1999, JRSSB)
- Bootstrapping
- Empirical information
- Numerical Differentiation

Example: Hidden Markov Model

- Hidden state $\mathbf{s} = (s_t)$, $t = 1, \ldots, T$, follows a Markov chain with initial state distribution π_k, $k = 1, \ldots, M$, and transition probability matrix $\{a_{k,l} = \Pr(s_{t+1} = l | s_t = k)\}$. (Note this matrix is row stochastic.)
- Given the state s_t, the observation u_t is independent of other observations and states
- Given state k, the observation follows distribution $b_k(u)$. For example, $b_k(u|\gamma_k)$ can be $N(\mu_k, \Sigma_k)$; or discrete.
- The observed data is $\mathbf{u} = (u_t)$, $t = 1, \ldots, T$.

Likelihood

- Parameters: θ contains π_k's, a_{kl}'s, γ_k's
- Missing (latent, unobserved) states: s_t, $t = 1, \ldots, T$.
- Observed data: u_t, $t = 1, \ldots, T$
- Complete data: $(\mathbf{s}, \mathbf{u})$, where $\mathbf{s} = (s_1, \ldots, s_T)$ and $\mathbf{u} = (u_1, \ldots, u_T)$.
- Complete data likelihood

$$\pi_{s_1} \prod_{t=2}^{T} a_{s_{t-1}, s_t} \prod_{t=1}^{T} b_{s_t}(u_t | \gamma_{s_t})$$

- Complete data log-likelihood

$$\log p(\mathbf{s}, \mathbf{u} | \theta) = \log \pi_{s_1} + \sum_{t=2}^{T} \log a_{s_{t-1}, s_t} + \sum_{t=2}^{T} \log b_{s_t}(u_t | \gamma_{s_t})$$

E-Step

Taking expectation of log $p(\mathbf{s}, \mathbf{u}|\theta')$ with respect to $p(\mathbf{s}|\mathbf{u}, \theta)$:

$$
\begin{aligned}
&E\bigg(\log p(\mathbf{s}, \mathbf{u})|\theta')|\mathbf{u}, \theta) \bigg) \\
&= \sum_{\mathbf{s}} p(\mathbf{s}|\mathbf{u}, \theta) \log p(\mathbf{s}, \mathbf{u}|\theta') \\
&= \sum_{k=1}^{M} L_k(1) \log \pi'_k + \sum_{t=2}^{T} \sum_{k=1}^{M} \sum_{l=1}^{M} H_{k,l}(t) \log a'_{k,l} \\
&\quad + \sum_{t=1}^{T} \sum_{k=1}^{M} L_k(t) \log b_{s_t}(u_t|\gamma'_k)
\end{aligned}
$$

where

$$L_k(t) = \Pr(s_t = k|\mathbf{u}) = \sum_{\mathbf{s}: s_t = k} p(\mathbf{s}|\mathbf{u})$$

and

$$H_{k,l}(t) = \Pr(s_t = k, s_{k+1} = l|\mathbf{u}) = \sum_{\mathbf{s}: s_t = k, s_{t+1} = l} \Pr(\mathbf{s}|\mathbf{u}).$$

M-step

If b_k's are multivariate normals, the M-step has closed-form solution:

$$\mu_k = \frac{\sum_{t=1}^{T} L_k(t) u_t}{\sum_{t=1}^{T} L_k(t)}$$

$$\Sigma_k = \frac{\sum_{t=1}^{T} L_k(t)(u_t - \mu_k)(u_t - \mu_k)^\top}{\sum_{t=1}^{T} L_k(t)}$$

$$a_{k,l} = \frac{\sum_{t=1}^{T-1} H_{k,l}(t)}{\sum_{t=1}^{T-1} L_k(t)}$$

$$\pi_k = \frac{\sum_{t=1}^{T} L_k(t)}{\sum_{k=1}^{M} \sum_{t=1}^{T} L_k(t)}$$

Forward and Backward Probability

- This is for efficient computation of $L_k(t)$'s and $H_{k,l}(t)$'s
- Computation of order M^2T and memory requirement of order MT.
- Forward probability

$$\alpha_k(t) = \Pr(u_1, \ldots, u_t, s_t = k)$$

- Backward probability

$$\beta_k(t) = \Pr(u_{t+1}, \ldots, u_T | s_t = k)$$

Forward and Backward Algorithms

- Forward probability recursion

$$\alpha_k(1) = \pi_k b_k(u_1), \qquad 1 \le k \le M,$$

$$\alpha_k(t) = b_k(u_t) \sum_{l=1}^{M} \alpha_l(t-1) a_{l,k}, \qquad 1 < t \le T, \quad 1 \le k \le M.$$

- Backward probability recursion

$$\beta_k(T) = 1, \qquad 1 \le k \le M,$$

$$\beta_k(t) = \sum_{l=1}^{M} a_{k,l} b_l(u_{t+1}) \beta_l(t+1), \qquad 1 \le t < T, \quad 1 \le k \le M.$$

Fast Algorithm for the E-step Quantities

$$
\begin{aligned}
L_k(t) &= \mathsf{Pr}(s_t = k|\mathbf{u}) = \frac{\alpha_k(t)\beta_k(t)}{p(\mathbf{u})} \\
H_{k,l}(t) &= \mathsf{Pr}(s_t = k, s_{k+1} = l|\mathbf{u}) = \frac{\alpha_k(t)a_{k,l}b_l(u_{t+1})\beta_l(t+1)}{p(\mathbf{u})} \\
p(\mathbf{u}) &= \sum_{k=1}^{M} \alpha_k(t)\beta_k(t), \qquad \forall t
\end{aligned}
$$

Majorization-Minimization Algorithm

Majorization-Minimization Algorithm

- The MM algorithm is not an algorithm, but a prescription for constructing optimization algorithms.
- The EM algorithm is a special case.
- An MM algorithm operates by creating a surrogate function that majorizes (minorizes) the objective function. When the surrogate function is optimized, the objective function is driven downhill (uphill).
- Majorization-Minimization, Minorization-Maximization.

Majorization-Minimization Algorithm

- It may generate an algorithm that avoids matrix inversion.
- It can linearize an optimization problem.
- It can conveniently deal with equality and inequality constraints.
- It can solve a non-differentiable problem by iteratively solving smooth problems.

Majorization-Minimization Algorithm

- A function $g(\theta \mid \theta^s)$ is said to majorize the function $f(\theta)$ at θ^s if
$$f(\theta^s) = g(\theta^s \mid \theta^s),$$
and
$$f(\theta) \le g(\theta \mid \theta^s)$$
for all θ.
- Let
$$\theta^{s+1} = \arg\min_{\theta} g(\theta \mid \theta^s).$$
- It follows that
$$f(\theta^{s+1}) \le g(\theta^{s+1} \mid \theta^s) \le g(\theta^s \mid \theta^s) = f(\theta^s).$$
The strict inequality holds unless $g(\theta^{s+1} \mid \theta^s) = g(\theta^s \mid \theta^s)$ and $f(\theta^{s+1}) = g(\theta^{s+1} \mid \theta^s)$.
- Therefore, by alternating between the majorization and the minimization steps, the objective function is monotonically decreasing and thus its convergence is guaranteed.

Majorization-Minimization Algorithm

- We can use any inequalities to construct the desired majorized/minorized version of the objective function. There are several typical choices:
  - Jensen's inequality: for a convex function $f(x)$,
    $$f(E(X)) \leq E(f(X)).$$
  - Convexity inequality: for any $\lambda \in [0, 1]$,
    $$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda)f(x_2)$$
  - Cauchy-Schwarz inequality.
  - Supporting hyperplanes.
  - Arithmetic-Geometric Mean Inequality.

Example: Penalized AFT model with induced smoothing

AFT model

- Let $\{T_i, C_i, \mathbf{x}_i\}$, $i = 1, \ldots, n$ be n independent copies of $\{T, C, \mathbf{X}\}$, where T_i and C_i are log-transformed failure time and log transformed censoring time, $\mathbf{x}_i$ is a $p \times 1$ covariate vector, and given $\mathbf{X}$, C and T are independent.
- The Accelerated failure time model has the form

$$T_i = \mathbf{x}_i^T \boldsymbol{\beta} + \epsilon_i, \qquad i = 1, \ldots, n.$$

- The observed data are $(Y_i, \delta_i, \mathbf{x}_i)$, where $Y_i = \min(T_i, C_i)$, $\delta_i = I(T_i < C_i)$.

AFT model

- For a case-cohort study, The weight-adjusted estimating equation with induced smoothing is

$$\tilde{U}_n(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\Delta_i(\mathbf{x}_i - \mathbf{x}_j)\Phi\left(\frac{e_j(\boldsymbol{\beta}) - e_i(\boldsymbol{\beta})}{r_{ij}}\right) = 0,$$

  where $e_j(\boldsymbol{\beta}) = Y_j - \mathbf{x}_j^{\mathrm{T}}\boldsymbol{\beta}$ and
  $r_{ij}^2 = n^{-1}(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j) = n^{-1}\|\mathbf{x}_i - \mathbf{x}_j\|^2$.
- Define $H(x) = x\Phi(x) + \phi(x)$, and

$$\tilde{L}_n(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\Delta_i r_{ij} H\left(\frac{e_j(\boldsymbol{\beta}) - e_i(\boldsymbol{\beta})}{r_{ij}}\right).$$

  Then solving the estimating equation $\tilde{U}_n(\boldsymbol{\beta}) = 0$ is equivalent to minimizing the objective function $\tilde{L}_n(\boldsymbol{\beta})$.
- The objective function $\tilde{L}_n(\boldsymbol{\beta})$ is convex in $\boldsymbol{\beta}$.

AFT model

- We propose the penalized AFT method, which amounts to minimize

$$\tilde{P}L_n(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\Delta_i r_{ij} H\left(\frac{e_j(\boldsymbol{\beta}) - e_i(\boldsymbol{\beta})}{r_{ij}}\right) + \lambda P(\boldsymbol{\beta}),$$

where $P(\boldsymbol{\beta})$ is some penalty function and λ is the tuning parameter.

AFT model

- To solve this problem, the key is to identity a surrogate function of $\tilde{L}_n(\boldsymbol{\beta})$ that is easy to work with. In view of the form of the objective function, an immediate idea is to majorize $H(x)$.
- Because $H(x)$ is twice differentiable and $H''(x) = \phi(x) < \phi(0) \approx 0.4$ for any $x \in \mathbb{R}$, it follows that for any $x^0 \in R$,

$$H(x \mid x^0) \le G(x \mid x^0) \equiv H(x^0) + (x - x^0)\Phi(x^0) + \frac{\phi(0)}{2}(x - x^0)^2.$$

- Note that in a standard optimization method based on quadratic approximation, e.g., Newton-Raphson, the $\phi(0)$ term is replaced with $\phi(x^0)$, which, however, does not guarantee the majorization.

AFT model

- Write

$$\tilde{L}_n(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} H(a_{ij} + \mathbf{z}_{ij}^{\mathrm{T}}\boldsymbol{\beta}) + \lambda P(\boldsymbol{\beta}),$$

  where $w_{ij} = \Delta_i r_{ij}$, $a_{ij} = (Y_j - Y_i)/r_{ij}$ and $\mathbf{z}_{ij} = (\mathbf{x}_i - \mathbf{x}_j)/r_{ij}$. Then

$$\begin{aligned}\tilde{L}_n(\boldsymbol{\beta} \mid \boldsymbol{\beta}^s) &\le \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} G(a_{ij} + \mathbf{z}_{ij}^{\mathrm{T}}\boldsymbol{\beta} \mid a_{ij} + \mathbf{z}_{ij}^{\mathrm{T}}\boldsymbol{\beta}^s) + \lambda P(\boldsymbol{\beta}) \\ &= \frac{1}{2}\boldsymbol{\beta}^{\mathrm{T}}\mathbf{A}\boldsymbol{\beta} - \mathbf{b}^{s^{\mathrm{T}}}\boldsymbol{\beta} + \lambda P(\boldsymbol{\beta}) + \text{const} \\ &\equiv \tilde{L}_n^*(\boldsymbol{\beta} \mid \boldsymbol{\beta}^s) + \text{const},\end{aligned}$$

  where

$$\mathbf{A} = \frac{\phi(0)}{n}\sum_{i=1}^{n}\sum_{i=1}^{n} w_{ij}\mathbf{z}_{ij}\mathbf{z}_{ij}^{\mathrm{T}}, \qquad \mathbf{b}^s = \mathbf{A}\boldsymbol{\beta}^s - \frac{1}{n}\sum_{i=1}^{n}\sum_{i=1}^{n} w_{ij}\Phi(a_{ij} + \mathbf{z}_{ij}^{\mathrm{T}}\boldsymbol{\beta}^s)\mathbf{z}_{ij}$$

- Minimizing $\tilde{L}_n^*(\boldsymbol{\beta} \mid \boldsymbol{\beta}^s)$ is a Lasso-type problem, for which a coordinate descent algorithm can be applied.

MM for Penalized AFT

```
Initialize $\boldsymbol{\beta}^0 \in \mathbb{R}^p$. Compute
    $$\mathbf{A} = \frac{\phi(0)}{n} \sum_{i=1}^{n} \sum_{i=1}^{n} w_{ij} \mathbf{z}_{ij} \mathbf{z}_{ij}^{\mathrm{T}}.$$
Set $k \leftarrow 0$.
repeat
    (1). Majorization step:
        $$\mathbf{b}^{k+1} \leftarrow \mathbf{A}\boldsymbol{\beta}^k - \frac{1}{n} \sum_{i=1}^{n} \sum_{i=1}^{n} w_{ij} \Phi(a_{ij} + \mathbf{z}_{ij}^{\mathrm{T}} \boldsymbol{\beta}^k) \mathbf{z}_{ij}.$$
    (2). Minimization step:
    Initialize $\tilde{\boldsymbol{\beta}}^0 = \boldsymbol{\beta}^k$.
    while $\|\tilde{\boldsymbol{\beta}}^s - \tilde{\boldsymbol{\beta}}^{s+1}\| / \|\tilde{\boldsymbol{\beta}}^s\| \geq \epsilon$ and $s < s_{max}$ do
        (i) For $j = 1, \ldots, p$,
            $$\tilde{\beta}_j^s \leftarrow \frac{1}{a_{jj}} \mathcal{S}(b_j^{k+1} - \mathbf{a}_j^{\mathrm{T}} \tilde{\boldsymbol{\beta}}^s + a_{jj} \tilde{\beta}_j^s, \lambda).$$
        (ii) $\tilde{\boldsymbol{\beta}}^{s+1} \leftarrow \tilde{\boldsymbol{\beta}}^s$.
        (iii)$s \leftarrow s + 1$.
    end while
        $$\boldsymbol{\beta}^{k+1} \leftarrow \tilde{\beta}^{s+1}.$$
    Set $k \leftarrow k + 1$.
until convergence, i.e., $\|\boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k\| / \|\boldsymbol{\beta}^k\| \leq \epsilon$.
```

Simulation Basics

Sampling Random Variables

The entire area of random sampling is built up the fact that we can use computing device to generate, deterministically, long sequences of numbers that by many accounts are very similar to long sequences of iid random variables following Unif$(0, 1)$.

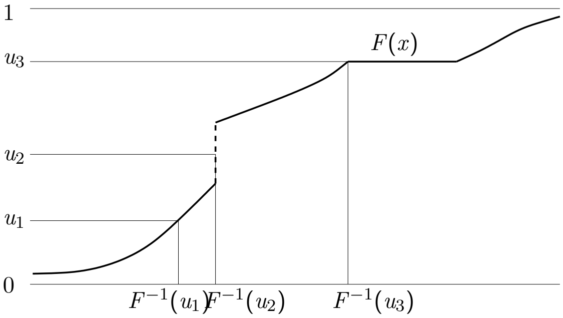
In our discussion, we assume that we can generate ideal sequence $U_1, U_2, \ldots$ iid $\sim$ Unif$(0, 1)$. The question is how to use them to simulate other random variables for vectors.

Inverse transform method

This method only applies to random variables taking values in $\mathbb{R}$. Let F be a distribution function. Recall that F is nondecreasing and right-continuous with $0 \le F(x) \le 1$. However, F may be discontinuous at some x and may be constant in certain intervals.
For $0 \le u \le 1$, define

$$F^{-1}(u) = \inf\{x : F(x) \ge u\}.$$

The inverse can be thought as follows. Plot the graph of F. If F has a jump at x, the graph of F will have a gap at x. Fill the gap with a vertical line. Draw a horizontal line at height u starting from $-\infty$. Then $F^{-1}(u)$ is the x-coordinate of the first point that the horizontal line meets with the gap-filled graph of F.



Theorem

If $U \sim$ Unif$(0,1)$, then $F^{-1}(U) \sim F$.

Proof.

For any x, $\mathbb{P}F^{-1}(U) \leq x = \mathbb{P}U \leq F(x) = F(x)$. □

Discrete Random Variable

Let X be a random variable taking values in $\{c_1, \ldots, c_n\}$. Note that c_i may not be sorted or even not real numbers. Let

$$q_0 = 0, \quad q_i = \sum_{j=1}^{i} \mathbb{P}\left\{X = c_j\right\}, \quad i = 1, \ldots, n$$

To sample X,

1. sample $U \sim \text{Unif}(0, 1)$;
2. find $K \in \{1, \ldots, n\}$, such that $q_{K-1} < U \leq q_K$;
3. set $X = c_K$. □

Exponential Variable

The exponential distribution with parameter θ, usually denoted by $\mathsf{Exp}(\theta)$, has

$$F(x) = 1 - e^{-x/\theta}, \quad x \geq 0.$$

Since $F^{-1}(u) = -\theta \ln(1-u)$ for $u \in (0,1)$,

$$-\theta \ln(1-U) \sim \mathsf{Exp}(\theta), \quad U \sim \mathsf{Unif}(0,1).$$

Since $1 - U \sim \mathsf{Unif}(0,1)$, there is also $-\theta \ln U \sim \mathsf{Exp}(\theta)$.

A standard Brownian motion $W(t)$ is a process with continuous sample paths that satisfies the following conditions

(i) $W(0) = 0$; and

(ii) for $0 \le t_1 < \cdots < t_k$, the increments

$$W(t_2) - W(t_1), \ldots, W(t_k) - W(t_{k-1})$$

are independent and each $W(t_i) - W(t_{i-1}) \sim N(0, t_i - t_{i-1})$.

Example Let T be the time at which a standard Brownian motion $W(t)$ attains its maximum over the time interval $[0, 1]$:

$$T = \text{inf}\{t \in [0, 1] : W(t) = \sup_{x \in [0,1]} W(x)\}.$$

Then T follows the *arcsine law* whose distribution is

$$F(x) = \frac{2}{\pi} \text{arcsin}(\sqrt{x}), \quad x \in [0, 1]$$

and hence

$$\frac{1 - \cos(U\pi)}{2} \sim T. \qquad \square$$

Proof.

For $x \in [0,1]$,

$$\begin{aligned}
\mathbb{P}\{T \le x\} &= \mathbb{P}\left\{\max_{t\in[0,x]} W(t) - W(x) \ge \max_{t\in[x,1]} W(t) - W(x)\right\} \\
&= \mathbb{P}\left\{\max_{t\in[-x,0]} W(t) \ge \max_{t\in[0,1-x]} W(t)\right\} \\
&= \mathbb{P}\left\{\max_{t\in[0,x]} \tilde{W}(t) \ge \max_{t\in[0,1-x]} W(t)\right\}
\end{aligned}$$

where $\tilde{W}$ is an independent copy of W. It is known (reflection principle) that $\max_{t\in[0,x]} W(t) \sim x|Z|$, where $Z \sim N(0,1)$. It is also known that if Z and $\tilde{Z}$ are iid $N(0,1)$, then $Z/\tilde{Z}$ is Cauchy with $F(x) = \pi^{-1} \arctan x + 1/2$. Therefore, $\mathbb{P}\{T \le x\} = \mathbb{P}\left\{(|Z/\tilde{Z}| \le \sqrt{x/(1-x)}\right\} = \frac{2}{\pi} \arctan \sqrt{x/(1-x)} = \frac{2}{\pi} \arcsin(\sqrt{x})$. □

Example Let

$$M = \sup_{t \in [0,1]} W(t)$$

Then, conditioning on $W(1) = b$, M follows the *Rayleigh distribution*

$$\mathbb{P}\{M \le x \,|\, W(1) = b\} = F_b(x) = 1 - e^{-2x(x-b)}, \quad x > \max(0, b).$$

Solving $u = 1 - e^{-2x(x-b)}$ gives

$$x = \frac{b \pm \sqrt{b^2 - 2\ln(1-u)}}{2}$$

Since $F_b(x) > 0$ only for $x > b$,

$$F_b^{-1}(u) = \frac{b + \sqrt{b^2 - 2\ln(1-u)}}{2}.$$

Then

$$\frac{b + \sqrt{b^2 - 2\ln U}}{2} \sim F_b.$$

□

Proof.

Let $T = \inf\{t : W(t) = x\}$. For $t \in (0, 1)$, $f_{T|W(1)}(t \,|\, b) \propto f_{W(1)|T}(b \,|\, t) f_T(t)$. By strong property of W,

$$f_{W(1)\,|\,T}(b \,|\, t) = \frac{e^{-(b-x)^2/2(1-t)}}{\sqrt{2\pi(1-t)}}.$$

and by scaling $\mathbb{P}\{T \le t\} = \mathbb{P}\left\{\max_{s\in[0,t]} W(s) \ge x\right\} = \mathbb{P}\left\{M \ge x/\sqrt{t}\right\}$.

□

Sampling a conditional distribution

Let $X \sim F$. To sample X conditional on $a < X \leq b$, recall

$$\mathbb{P}\{a < X \leq b\} = F(b) - F(a).$$

In order for the sampling to make sense, assume $F(a) < F(b)$. Let $A = F(a)$ and $B = F(b)$. Then

$$F^{-1}(A + (B - A)U)$$

follows the conditional distribution, since

$$\begin{aligned}\mathbb{P}\left\{F^{-1}(A + (B - A)U) \leq x\right\} &= \mathbb{P}\{A + (B - A)U \leq F(x)\} \\ &= \mathbb{P}\left\{U \leq \frac{F(x) - F(a)}{F(b) - F(a)}\right\} = \frac{F(x) - F(a)}{F(b) - F(a)}.\end{aligned}$$

The right hand side is just $\mathbb{P}\{X \leq x \,|\, a < X \leq b\}$.

Example Let $c > 0$ and $X \sim \text{Exp}(\theta)$. To sample X conditional on $X \geq c$, recall the exponential distribution is memoryless in the sense that for any $c > 0$,

$$X - c \sim \text{Exp}(\theta) \quad \text{given} \quad X > c.$$

Therefore, the conditional distribution can be sampled by $-\theta \ln U + c$ with $U \sim \text{Unif}(0, 1)$. □

Numerical evaluation of F^{-1}

If F^{-1} is not known explicitly, the inverse transform method is still applicable through numerical evaluation of F^{-1}. If F is continuous, computing $F^{-1}(u)$ is equivalent to finding a root x of the equation $F(x) - u = 0$. For a distribution F with density f, Newton's method procedures a sequence

$$x_{n+1} = x_n - \frac{F(x_n) - u}{f(x_n)}$$

iteratively, given a starting point x_0. Under suitable conditions, $x_n \to F^{-1}(u)$.

R provides many sampling functions. In each of the following functions, `n` is the number of observations to be sampled.

- `runif(n, min=0, max=1)`: Unif(a, b);
- `rnorm(n, mean=0, sd=1)`: $N(\mu, \sigma^2)$;
- `rpois(n, lambda)`: Poisson(λ)

  $$p_n = e^{-\lambda}\lambda^n/n!.$$

- `rbeta(n, shape1, shape2)`: Beta(α, β)

  $$f(x) = x^{\alpha-1}(1-x)^{\beta-1}/B(\alpha, \beta), \quad 0 < x < 1;$$

  note α and β must be positive.
- `rgamma(n, shape, rate=1)`: Gamma$(\alpha, 1/r)$

  $$f(x) = r^{\alpha}x^{\alpha-1}e^{-rx}/\Gamma(\alpha). \quad x > 0;$$

  note α must be positive.

Rejection sampling

The method was introduced by Von Neumann and is among the most widely applicable mechanisms. In standard statistical software, the method is implemented to randomly sample many important distributions, such as normal distributions, Gamma distributions, Beta distributions, Poisson distributions. Much more detail on specific developments can be found in the following books.

1. Wolfgang Hörmann, Josef Leydold, and Gerhard Derflinger, *Automatic nonuniform random variate generation*, Springer-Verlag, 2004.
2. Luc Devroye, *Nonuniform random variate generation*, Springer-Verlag, 1986.

Let $\mathscr{X}$ be a discrete set or a Euclidean space. Suppose the goal is to sample from a target density f defined on $\mathscr{X}$. Suppose we know that $f(x)$ is proportional to a function $q(x)$, i.e.,

$$f(x) = q(x)/C, \quad x \in \mathscr{X}.$$

In order for f to be a probability density, C must satisfy

$$C = \begin{cases} \int_{\mathscr{X}} q(x)\,\mathrm{d}x & \text{if } \mathscr{X} \text{ is continuous} \\ \sum_{x\in\mathscr{X}} q(x) & \text{if } \mathscr{X} \text{ is discrete} \end{cases}$$

Note that C is often intractable, especially when $\mathscr{X}$ or q is complicated.

Let g be an "instrumental" density on $\mathscr{X}$ from which we know how to generate samples (easily). Suppose g has the property that for some constant $\alpha > 0$,

$$q(x) \leq \alpha g(x), \qquad \text{for all } x \in \mathscr{X}.$$

The function $\alpha g(x)$ is known as an *envelope*.

Rejection sampling

1. Sample $X \sim g$ and $U \sim \text{Unif}(0, 1)$
2. If
$$U > \frac{q(X)}{\alpha g(X)}$$
then go to Step 1; otherwise return X

The returned value is a random sample from the density $f(x)$.

Even when the complete form of $f(x)$ is known, it is often still advantageous to use rejection sampling, by simply using $q(x) = f(x)$.

As can be seen, the number of iterations until a sample is accepted follows a geometric distribution, and in each iteration, the probability of acceptance is

$$p_a = \frac{1}{\alpha} \int_{\mathscr{X}} q(x) \, \mathrm{d}x.$$

To reduce the expected number of iterations, p_a should be large. Equivalently, α should be small. Because α must satisfy $q(x) \le \alpha g(x)$, given $g(x)$, the smallest possible α is

$$\alpha = \sup \frac{q(x)}{g(x)}.$$

To further reduce α, one has to choose appropriate g. Of course, one has to make sure g is easy to sample at the same time.

Sometimes, even though the instrument density g is easy to sample, it is tedious to write down the complete form of $g(x)$. Suppose $g(x) = Ch(x)$, where C is the known (but complicated) normalizing constant. Then the algorithm can be modified as follows. Let β be a constant such that

$$\frac{q(x)}{h(x)} \leq \beta$$

for all x with $q(x) > 0$. (Actually, $\beta = \alpha / C$.) Then in Step 2, replace $q(X)/\alpha g(X)$ with

$$\frac{q(X)}{\beta h(X)}.$$

Sometimes it is easier to express $f(x)$ as a function proportional to the sum of a set of functions,

$$f(x) \propto \sum_{k=1}^{m} q_k(x).$$

Superpose that we can find densities $g_1(x)$, ..., $g_m(x)$ that are easy to sample, and constants $\alpha_1, \dots, \alpha_m$, such that

$$q_k(x) \leq \alpha_k g_k(x).$$

Then the rejection sampling can be modified as follows. Note that the complete form of each $g_k(x)$ has to be known.

Rejection sampling

1. Sample k from $\{1, \ldots, m\}$ with probabilities $p_k \propto \alpha_k$.
2. Sample $X \sim g_k$ and $U \sim$ Unif$(0, 1)$
3. If
$$U > \frac{q_k(X)}{\alpha_k g_k(X)}$$
then go to Step 1; otherwise return X

The returned value is a random sample from the density $f(x)$.

Proof: Let Y be a sampled value. To show that Y has density f, it suffices to show that for any set $A \subset \mathscr{X}$, $\mathbb{P}\{Y \in A\} = \int_A f(x)\,\mathrm{d}x$. From the procedure,

$$\begin{aligned}\mathbb{P}\{Y \in A\} &= \mathbb{P}\{X \in A \,|\, X \text{ is accepted}\} \\ &= \frac{\mathbb{P}\{X \in A \text{ and } X \text{ is accepted}\}}{\mathbb{P}\{X \text{ is accepted}\}}.\end{aligned}$$

Let $C = \alpha_1 + \cdots + \alpha_m$. Then

$$\begin{aligned}&\mathbb{P}\{X \in A \text{ and } X \text{ is accepted}\} \\ &= \sum_{k=1}^m p_k \mathbb{P}\{X \in A \text{ and } X \text{ is accepted} \,|\, X \text{ is from } g_k\} \\ &= C^{-1} \sum_{k=1}^m \alpha_k \mathbb{P}\{X \in A \text{ and } X \text{ is accepted} \,|\, X \text{ is from } g_k\}\end{aligned}$$

For each k, let

$$r(x) = \frac{q_k(x)}{\alpha_k g_k(x)}.$$

Then

$$\begin{aligned}&\mathbb{P}\left\{X \in A \text{ and } X \text{ is accepted} \,|\, X \text{ is from } g_k\right\}\\=&\mathbb{P}\left\{X \in A, U \le r(X) \,|\, X \text{ is from } g_k\right\}\end{aligned}$$

Given $X = x$, the probability that $U \le r(X)$ is simply $r(x)$. Therefore,

$$\begin{aligned}\mathbb{P}\left\{X \in A, U \le r(X) \,|\, X \text{ is from } g_k\right\} &= \int \mathbf{1}\left\{x \in A\right\} r(x) g_k(x) \, \mathrm{d}x\\&= \int_A \frac{q_k(x)}{\alpha_k g_k(x)} g_k(x) \, \mathrm{d}x\\&= \frac{1}{\alpha_k} \int_A q_k(x) \, \mathrm{d}x.\end{aligned}$$

Note the factor $1/\alpha_k$. As a result,

$$\mathbb{P}\{X \in A \text{ and } X \text{ is accepted}\} = C^{-1} \sum_{k=1}^{m} \int_A q_k(x)\,\mathrm{d}x.$$

In particular, applying the formula to $A = \mathscr{X}$,

$$\mathbb{P}\{X \text{ is accepted}\} = C^{-1} \sum_{k=1}^{m} \int q_k(x)\,\mathrm{d}x,$$

As a result,

$$\mathbb{P}\{Y \in A\} = \frac{\sum_{k=1}^{m} \int_A q_k(x)\,\mathrm{d}x}{\sum_{k=1}^{m} \int q_k(x)\,\mathrm{d}x} = \int_A f(x)\,\mathrm{d}x.$$

Example: Beta Variable

For $a, b > 0$, Beta(a, b) has a density

$$f(x) = \begin{cases} \frac{x^{a-1}(1-x)^{b-1}}{B(a,b)}, & 0 \le x \le 1 \\ 0 & \text{elsewhere} \end{cases}$$

and can be sampled by using the fact for independent $Y \sim$ Gamma$(a, 1)$ and $Z \sim$ Gamma$(b, 1)$,

$$\frac{Y}{Y+Z} \sim \text{Beta}(a, b).$$

One can try some easy alternatives without sampling Gamma distributions.

Method 1

This method works only when $a, b \geq 1$. If $a = b = 1$, then Beta(a, b) is simply Unif$(0, 1)$. Let $a > 1$ and $b > 1$. Then $f(x)$ is maximized

$$x_0 = \frac{a-1}{a+b-2}.$$

Choose g to be the uniform density on $[0, 1]$. Then the optimal α is

$$\alpha = \sup \frac{f(x)}{g(x)} = f(x_0).$$

The algorithm is implemented as follows.

- Draw X, $U \sim$ Unif$(0, 1)$ until $f(x_0)U \leq f(X)$ and then return X.

Method 2

Let $U \sim \text{Unif}(0,1)$. Observe that for any a, $b > 0$,

$$U^{1/a} \sim \text{Beta}(a,1), \quad 1 - U^{1/b} \sim \text{Beta}(1,b).$$

The densities of the distributions are

$$f_a(x) = ax^{a-1}, \quad f_b(x) = b(1-x)^{b-1}.$$

Therefore, if $g(x)$ is a mixture of $f_a(x)$ and $f_b(x)$,

$$g(x) = (1-\lambda)f_a(x) + \lambda f_b(x),$$

where $\lambda \in (0,1)$, then $f(x)/g(x)$ is bounded.
To sample from $g(x)$, one can run the following steps.

1. Sample U and $V \sim \text{Unif}(0,1)$
2. If $V > \lambda$, then return $X = U^{1/a}$; otherwise return $X = 1 - U^{1/b}$. □

Conditional distributions

Let A be a subset of $\mathscr{X}$. To sample X conditional on $X \in A$ one can use the following crude rejection sampling procedure

- Sample X until $X \in A$ and then return X.

However, sometimes more carefully designed rejection sampling is needed to make efficient sampling.

Example: Sampling from Normal Tails

Let $c > 0$. When c is large, to sample X from $N(0, 1)$ conditional on $X \geq c$, the simple rejection sampling is very inefficient. To improve the efficiency, an exponential distribution can be used as the envelope.
It suffices to sample $X - c$. Given $X \geq c$, the conditional density of $X - c$ for $X \sim N(0, 1)$ is

$$f(x) = \frac{\phi(x + c)}{1 - \Phi(c)}, \quad x \geq 0,$$

where $\phi(x)$ is the standard normal density.
On the other hand, for $X \sim \text{Exp}(\lambda)$, the conditional density of $X - c$ is

$$g(x) = \lambda e^{-\lambda x}.$$

Given λ, the optimal

$$\alpha = \sup_{x \geq 0} \frac{\phi(x+c)/[1-\Phi(c)]}{\lambda e^{-\lambda x}} = \frac{\exp(\frac{1}{2}\lambda^2 - \lambda c)}{\sqrt{2\pi}\lambda[1-\Phi(c)]}.$$

The value of λ that minimizes the last expression is

$$\lambda = \frac{c + \sqrt{c^2+4}}{2}.$$

Then the optimal

$$p_a = \frac{1}{\alpha} = \frac{\sqrt{\pi}(c + \sqrt{c^2+4})[1-\Phi(c)]}{\sqrt{2e}}.$$

□

Gamma distributions

For r, $\lambda > 0$, $\text{Gamma}(r, \lambda)$ has a density

$$f(x) = \frac{x^{r-1} e^{-x/\lambda}}{\lambda^r \Gamma(r)}, \quad x > 0.$$

1. $\text{Exp}(\lambda) = \text{Gamma}(1, \lambda)$. If $U \sim \text{Unif}(0, 1)$, then $-\lambda \ln U \sim \text{Exp}(\lambda)$.
2. $\chi^2_k = \text{Gamma}(\frac{k}{2}, \frac{1}{2})$.
3. If $X \sim \text{Gamma}(r, \lambda)$ and $c > 0$, then $cX \sim \text{Gamma}(r, c\lambda)$.
4. If $X_i \sim \text{Gamma}(r_i, \lambda)$, $i = 1, \ldots, n$, are independent, then

$$X_1 + \cdots + X_n \sim \text{Gamma}(r_1 + \cdots + r_n, \lambda).$$

Some simple ways to sample special Gamma distributions.

1. If r is an integer, then Gamma(r, λ) can be sampled by

$$-\lambda \sum_{i=1}^{r} \ln U_i$$

   where $U_1, \ldots, U_r$ are iid $\sim$ Unif$(0, 1)$

2. If r is an half integer $n/2$, then Gamma(r, λ) can be sampled by

$$2\lambda \sum_{i=1}^{n} X_i^2$$

   where $X_1, \ldots, X_n$ are iid $\sim N(0, 1)$.

Note that the above methods take longer time as r increases, because they require sampling of more and more variables.

There are many rejection sampling algorithms for Gamma distributions Gamma$(r, 1)$ that are *universally fast*. That is, for such an algorithm, one can find a constant $M > 0$, such that for any $r > 0$, the expected number of iterations is no greater than M.

Typically, the cases $r \geq 1$ and $r < 1$ are treated separately. For $r \geq 1$, one universally fast algorithm, which is due to Best (1978), is as follows. Let

$$b = r - 1, \quad c = 3r - 3/4.$$

1. Draw U, V iid $\sim$ Unif$(0, 1)$ and set $W = U(1 - U)$, $Y = \sqrt{c/W}(U - 1/2)$, $X = b + Y$.
2. Set $Z = 64 W^3 V^2$. If $X \geq 0$, and either $Z \leq 1 - 2Y^2/X$ or $\ln(Z) \leq 2[b \ln(X/b) - Y]$, then return X; otherwise go to Step 1.

For $r \leq 1$, the following algorithm of Ahrens and Dieter (1974) is universally fast. Let

$$p = \frac{e}{r + e}$$

and the instrument density

$$g(x) = \begin{cases} prx^{r-1}, & x \in [0, 1] \\ (1 - p)e^{-x+1} & x > 1. \end{cases}$$

That is, g is a mixture of Beta$(r, 1)$ and Exp(1) conditional on $(1, \infty)$. Then keep sampling $X \sim g$ and $U \sim$ Unif$(0, 1)$ until $U \leq f(X)/[Cg(X)]$, where $f(X) = x^{r-1}e^{-x}/\Gamma(a)$ is the density of Gamma$(r, 1)$ and C can be set equal to 1.39. In principle, the expected number of iterations is no greater than 1.39.

Box–Müller method.

Let Φ be the distribution function of $N(0,1)$. To sample $X \sim N(0,1)$, a straightforward method is to apply $\Phi^{-1}(U)$ with $U \sim \text{Unif}(0,1)$. There are two other often used methods.

This method does not need rejection sampling. However, the computation of sine and cosine functions can be time consuming.

1. Sample U_1, U_2 iid $\sim \text{Unif}(0,1)$
2. Set $R = \sqrt{-2 \ln U_1}$
3. Return

$$Z_1 = R \cos(2\pi U_2), \quad Z_2 = R \sin(2\pi U_2).$$

Then Z_1, Z_2 are iid $\sim N(0,1)$.

Proof: For any $r > 0$,

$$\begin{aligned}\mathbb{P}\{R \le r\} &= \mathbb{P}\left\{-2 \ln U_1 \le r^2\right\} \\ &= \mathbb{P}\left\{U_1 \ge e^{-r^2/2}\right\} = 1 - e^{-r^2/2}\end{aligned}$$

So R has density $re^{-r^2/2}$. Recall that if $\boldsymbol{X}$ is a random vector of d dimension and has a density $f(\boldsymbol{x})$, while

$$\boldsymbol{\phi}(\boldsymbol{x}) = (\phi_1(\boldsymbol{x}), \ldots, \phi_d(\boldsymbol{x}))$$

is a 1-to-1 differentiable function, then $\boldsymbol{Y} = \boldsymbol{\phi}(\boldsymbol{X})$ is a random vector with density

$$g(\boldsymbol{y}) = \frac{f(\boldsymbol{x})}{|\det[\boldsymbol{\phi}'(\boldsymbol{x})]|}, \text{ with } \boldsymbol{x} = \boldsymbol{\phi}^{-1}(\boldsymbol{y}).$$

Let $\boldsymbol{X} = (R, U_2)$ and

$$\boldsymbol{\phi}(\boldsymbol{x}) = (r \cos(2\pi u), r \sin(2\pi u)).$$

Then $\boldsymbol{Y} = (Z_1, Z_2) = \boldsymbol{\phi}(\boldsymbol{X})$.

First, $\boldsymbol{X} = (R, U_2)$ has a joint density $f(r, u) = re^{-r^2/2}$. Second,

$$\boldsymbol{\phi}'(\boldsymbol{x}) = \begin{pmatrix} \cos(2\pi u) & -2\pi r \sin(2\pi u) \\ \sin(2\pi u) & 2\pi r \cos(2\pi u) \end{pmatrix}$$

and so $|\det[\boldsymbol{\phi}'(\boldsymbol{x})]| = 2\pi r$. So $\boldsymbol{Y} = (Z_1, Z_2)$ has density

$$g(z_1, z_2) = \frac{re^{-r^2/2}}{2\pi r} = \frac{e^{-r^2/2}}{2\pi},$$

with (r, u) satisfying

$$z_1 = r\cos(2\pi u_2), \quad z_2 = r\sin(2\pi u_2).$$

Then $r^2 = z_1^2 + z_2^2$ and so

$$g(z_1, z_2) = \frac{e^{-(z_1^2+z_2^2)/2}}{2\pi},$$

showing (Z_1, Z_2) are iid $\sim N(0, 1)$.

Marsaglia–Bray method.

Let D be the disk centered at $(0,0)$ with radius 1. Then the method goes as follows:

1. use rejection sampling to sample (X, Y) from Unif(D)
2. set $R = \sqrt{X^2 + Y^2}$ and

$$T = \frac{2\sqrt{-\ln R}}{R}$$

3. return $Z_1 = TX$, $Z_2 = TY$

Again, Z_1 and Z_2 are iid $\sim N(0,1)$.

Multivariate normal distributions

Let $\boldsymbol{\mu} \in \mathbb{R}^d$ and $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$ be nonnegative definite. Recall that for $\boldsymbol{X} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $\boldsymbol{A} \in \mathbb{R}^{n\times d}$,

$$\boldsymbol{A}\boldsymbol{X} \sim N(\boldsymbol{A}\boldsymbol{\mu}, \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^t).$$

Based on this, to sample $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, a method is to first compute the *Cholesky factorization* of $\boldsymbol{\Sigma}$, i.e.,

$$\boldsymbol{\Sigma} = \boldsymbol{A}\boldsymbol{A}^t,$$

with $\boldsymbol{A}$ being lower triangular, and then sample

$$\boldsymbol{\mu} + \boldsymbol{A}\boldsymbol{Z},$$

with all the coordinates of $\boldsymbol{Z}$ being iid $\sim N(0, 1)$.

Conditioning formula

Let $\boldsymbol{X}_1 \in \mathbb{R}^m$, $\boldsymbol{X}_2 \in \mathbb{R}^n$ be jointly normal, such that

$$\mathbb{E}\boldsymbol{X}_i = \boldsymbol{\mu}_i, \quad \mathrm{Cov}(\boldsymbol{X}_i, \boldsymbol{X}_j) = \boldsymbol{\Sigma}_{ij}.$$

If $\boldsymbol{\Sigma}_{22}$ is of full rank, then, conditional on $\boldsymbol{X}_2 = \boldsymbol{x}$, $\boldsymbol{X}_1 \sim N(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$, where

$$\begin{aligned} \boldsymbol{\mu}_c &= \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_2), \\ \boldsymbol{\Sigma}_c &= \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}. \end{aligned}$$

Note that $\boldsymbol{\Sigma}_c$ is independent of $\boldsymbol{x}$.

Squeezed rejection sampling

Recall that in the ordinary rejection sampling of $f(x)$, we have $q(x)$, $g(x)$, and α available, such that $f(x) \propto q(x)$, where the proportionality factor may be intractable, and $g(x)$ a density function such that $q(x) \leq \alpha g(x)$. Every time $X \sim g$ is drawn, $q(X)$ needs to be evaluated. If the evaluation is computationally costly, it slows down the simulation.

1. Squeezed rejection sampling can be used to preempt the evaluation of $q(X)$, hence improving simulation speed.
2. The sampling is still exact.

Choose a *squeezing function* s that is easy to compute, while satisfying

$$s(x) \leq q(x) \text{ for all } x$$

Squeezed rejection sampling

1. Sample $X \sim g$, $U \sim$ Unif$(0,1)$, and compute $z = \alpha Ug(X)$
2. If $z \le s(X)$, then return X
   otherwise, if $z \le q(X)$, then return X
   otherwise, go to Step 1.

In contrast, the ordinary rejection sampling directly goes from Step 1 to 3. Because of Step 2, the proportion of iterations in which evaluation of q is avoided is

$$\mathbb{P}\left\{ U \le \frac{s(X)}{\alpha g(X)} \right\} = \frac{1}{\alpha} \int s(x)\, \mathrm{d}x.$$

Adaptive rejection sampling

The method developed by Gilks and Wild works well for density functions on an interval of $\mathbb{R}$ which are continuous, differentiable, and *log-concave*. Log-concave densities are common in applications, for example, all $N(\mu, \sigma^2)$, Gamma(α, β) with $\alpha \geq 1$, and Beta(α, β) with α, $\beta \geq 1$ are log-concave.

Suppose the density f is a density defined on (a, b), where $-\infty \leq a < b \leq \infty$, such that

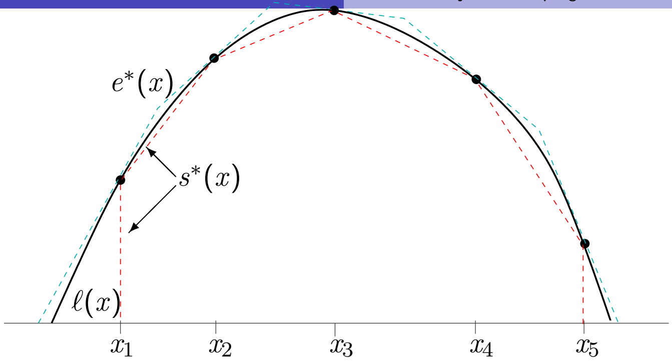
$$f(x) \propto q(x) = e^{\ell(x)}$$

where $\ell(x)$ is concave. The idea of the method is to construct two piecewise linear functions $e^*(x)$ and $s^*(x)$, such that

$$s^*(x) \leq \ell(x) \leq e^*(x)$$

Then $e(x) = \exp(e^*(x))$ and $s(x) = \exp(s^*(x))$ are envelope and squeezing functions respectively,

$$s(x) \leq q(x) \leq e(x).$$



The two piecewise linear functions s^* and e^* are shown above. First, select a set of points $a < x_1 < \cdots < x_n < b$. Define $s^*(x)$ to be $-\infty$ outside $[x_1, x_n]$ and to be linear on each $[x_i, x_{i+1}]$, such that its graph is the line segment connecting $(x_i, \ell(x_i))$ and $(x_{i+1}, \ell(x_{i+1}))$. To construct $e^*(x)$, draw tangent lines to the graph of ℓ at each $(x_i, \ell(x_i))$ and cut off the parts that are beyond the intersection points of the tangent lines. Since $\ell(x)$ is concave, it is guaranteed that $s^*(x) \le \ell(x) \le e^*(x)$. The "corner" points of $e^*(x)$ can be derived in close form. We omit the detail for brevity.

Clearly, $s(x) = \exp(s^*(x))$ can be used directly as a squeezing function. On the other hand, the $e(x)$ is not a density function. Instead, letting

$$C = \int e(x)\,\mathrm{d}x,$$

$g(x) = e(x)/C$ is a density function and satisfies $q(x) \le Cg(x)$. The rest of the method is then the standard squeezed rejection sampling.

To see why the method often works so well, note the following.

- First, $g(x)$ is an exponential function on each of the intervals $(-\infty, x_1)$, $[x_1, x_2]$, $\dots$, $[x_{n-1}, x_n]$, (x_n, ∞). Therefore, it can be sampled using the sampling for exponential distributions, which is easy.
- Second, C can be evaluated easily as well, again because $e(x)$ is piecewise exponential. Third, since $s^*(x)$ is piecewise linear, it is very easy to evaluate. As a result, $s(x) = \exp(s^*(x))$ is also easy to evaluate.

Sampling importance resampling (SIR)

Like rejection sampling, SIR uses an auxiliary density g, this time called an importance sampling function. After getting a (large) sample from g, it is resampled according to appropriate weights. This *approximately* generates a sample from the target density f.

The weights are called the standardized importance weights. Given $X_1, \ldots, X_m \sim g$, the weight for X_i is $w(X_i) \propto f(X_i)/g(X_i)$, specifically,

$$w(X_i) = \frac{f(X_i)/g(X_i)}{\sum_{k=1}^{m} f(X_k)/g(X_k)}.$$

- If $f = Cq$ for some unknown $C > 0$ but known q, the same $w(X_i)$ can be computed with $f(X_i)$ being replaced by $q(X_i)$.

SIR

Fix m and n.

1. Sample $X_1, \dots, X_m$ iid from g.
2. Calculate $w(X_1)$, …, $w(X_m)$.
3. Draw n draws with replacement from $X_1, \dots, X_m$ with probabilities $w(X_1)$, …, $w(X_m)$.

- The returned values $Y_1, \dots, Y_n$ consist a random sample *approximately* following the density $f(x)$.
- In order for $Y_1, \dots, Y_n$ to behave like an iid sample from $f(x)$, there should be $m \gg 1$ and $n/m \to 0$ as $m \to \infty$.

Let $h(x)$ be a function proportional to $f(x)/g(x)$. The above SIR is a procedure to sample from

$$f(x) \propto h(x)g(x).$$

It can be extended to

$$f(x) \propto \sum_{k=1}^{K} h_k(x)g_k(x)$$

where $g_k(x)$ are probability densities and $h_k(x) \geq 0$ positive functions. However, here the complete form of $g_k(x)$ has to be known. The procedure can be modified as follows.

SIR

Fix m and n.

1. For $k = 1, \ldots, K$, sample X_{k1}, . . . , X_{km} iid from g_k.
2. Calculate the weights
$$w_{kj} \propto h_k(X_{kj})$$
for $k = 1, \ldots, K$, $j = 1, \ldots, m$, such that their total is 1.
3. Make n draws with replacement from $\{X_{kj}\}$ with probabilities w_{kj}.

To justify SIR, we show that when $m \gg 1$, Y_1 (or Y_2, Y_3, etc) has a distribution approximately equal to f. Let

$$C = \sum_{k=1}^{K} \int h_k(x) g_k(x) \, \mathrm{d}x.$$

Then

$$f(x) = C^{-1} \sum_{k=1}^{K} h_k(x) g_k(x).$$

We need to show that, if A is an arbitrary region, then, as $m \to \infty$,

$$\mathbb{P}\{ Y_1 \in A \} \to \int_A f(x) \, \mathrm{d}x.$$

Denote by $\boldsymbol{X}$ the set of X_{kj}, $k = 1, \dots, K$, $j = 1, \dots, m$. *Given* $\boldsymbol{X}$, in the resampling step, the probability that X_{kj} is drawn to be Y_1 is

$$w_{kj} = \frac{h_k(X_{kj})}{\sum_{k=1}^K \sum_{j=1}^m h_k(X_{kj})}.$$

Then

$$\begin{aligned}
\mathbb{P}\{Y_1 \in A \,|\, \boldsymbol{X}\} &= \sum_{k=1}^K \sum_{j=1}^m \mathbb{P}\{X_{kj} \text{ is drawn to be } Y_1\} \mathbf{1}\{X_{kj} \in A\} \\
&= \frac{\dfrac{1}{m} \displaystyle\sum_{k=1}^K \sum_{j=1}^m \mathbf{1}\{X_{kj} \in A\} h_k(X_{kj})}{\dfrac{1}{m} \displaystyle\sum_{k=1}^K \sum_{j=1}^m h_k(X_{kj})}.
\end{aligned}$$

By the Strong Law of Large Numbers, with probability 1, as $m \to \infty$, for each k

$$\frac{1}{m} \sum_{j=1}^{m} \mathbf{1}\left\{X_{kj} \in A\right\} h_k(X_{kj}) \to \int \mathbf{1}\left\{x \in A\right\} h_k(x) g_k(x) \,\mathrm{d}x$$

and so

$$\begin{aligned} &\frac{1}{m} \sum_{k=1}^{K} \sum_{j=1}^{m} \mathbf{1}\left\{X_{kj} \in A\right\} h_k(X_{kj}) \\ &\to \int \mathbf{1}\left\{x \in A\right\} \sum_{k=1}^{K} h_k(x) g_k(x) \,\mathrm{d}x = C \int \mathbf{1}\left\{x \in A\right\} f(x) \,\mathrm{d}x \end{aligned}$$

In particular, let A be the entire domain of possible values of X_{jk}. Then

$$\frac{1}{m}\sum_{k=1}^{K}\sum_{j=1}^{m} h_k(X_{kj}) \to C.$$

So with probability one, as $m \to \infty$,

$$\mathbb{P}\{\, Y_1 \in A \mid \boldsymbol{X} \} \to \int_A f(x)\,\mathrm{d}x.$$

Since we are only interested in the distribution of Y_1, we need to remove the dependence on $\boldsymbol{X}$. This requires finding the limit of $\mathbb{P}\{\, Y_1 \in A\}$ as $m \to \infty$. Note

$$\mathbb{P}\{\, Y_1 \in A\} = \mathbb{E}[\mathbb{P}\{\, Y_1 \in A \mid X_1, \ldots, X_m\}].$$

By the Dominated Convergence Theorem,

$$\lim_{m\to\infty} \mathbb{P}\{\, Y_1 \in A\} = \mathbb{E}\left[\lim_{m\to\infty} \mathbb{P}\{\, Y_1 \in A \mid \boldsymbol{X}\}\right] = \mathbb{E}\left[\int_A f(x)\,\mathrm{d}x\right] = \int_A f(x)\,\mathrm{d}x$$

Sampling using Markov chains

The sampling using Markov chains is a very important method. It is highly adaptive to various difficult sampling problems, especially high-dimensional problems. Although in most applications it is an approximate sampling method, it often has good performance. Another advantage of the method is the ease of its implementation.
The method is often used to estimate $\mathbb{E}[h(X)]$ for a function $h(X)$ of random variable X. In this context, the method is known as Markov chain Monte Carlo (MCMC). Here we will use this term, but only consider the sampling aspect.

Let $\mathscr{X}$ be a discrete or Euclidean space. Recall that a stochastic process $\{X_t, t = 0, 1, 2, \dots\}$ taking values $\mathscr{X}$ is called a Markov chain if

$$\mathbb{P}\{X_t \in A \mid X_{t-1}, \dots, X_1, X_0\} = \mathbb{P}\{X_t \in A \mid X_{t-1}\} \text{ a.s.}$$

for all $t > 0$ and (measurable) $A \subset \mathscr{X}$. The marginal distribution of X_0 is called the initial distribution of of the Markov chain (X_t). If every X_t has a conditional density given X_s, $0 \le s < t$, then the condition can be written as

$$p(x_t \mid x_{t-1}, \dots, x_1, x_0) = p(x_t \mid x_{t-1}).$$

The chain is homogeneous if for any a and b,

$$p(x_t = b \mid x_{t-1} = a)$$

is the same for all $t > 0$. Denote by $q(b \mid a)$ to be the conditional density and call $q(\cdot \mid \cdot)$ the transition kernel of the Markov chain.

A homogeneous Markov chain $(X_t, t = 0, 1, 2, \ldots)$ in general is not stationary, because the density p_t of X_t is given by

$$p_t(x) = \int q(x \,|\, y) p_{t-1}(y) \, \mathrm{d}y$$

and hence is not necessarily the same as p_{t-1}. However, if X_0 follows a distribution π that satisfies

$$\pi(x) = \int q(x \,|\, y) \pi(y) \, \mathrm{d}y,$$

then $p_t \equiv \pi$ for all $t \geq 0$. This then implies the stationarity of (X_t). Thus π is called a *stationary* distribution of (X_t).

Importantly, under certain conditions, no matter the initial distribution p_0, as $t \to \infty$, $p_t \to \pi$. The existence and uniqueness of stationary distribution, and the convergence of p_t are true in a large number of cases, in particular, when $\mathscr{X}$ is finite and $p(y \,|\, x) > 0$ for all x, $y \in \mathscr{X}$.

The idea of MCMC is to construct a Markov chain (X_t) such that it has the target distribution f as the unique stationary distribution. If this can be done, then one has a good reason to expect that X_t is a random sample from a distribution very close to f, as long as n is large enough. Generally speaking, it is relatively easy to specify the transition kernel of a Markov chain, however, it is much harder to determine the corresponding stationary distribution(s). So a basic question is, how to choose $q(\cdot \mid \cdot)$ for (X_t), such that f is *a* stationary distribution of (X_t)? If this question is answered, then one can proceed to find conditions to make sure that (X_t) has a unique stationary distribution and p_t converges to it.

Suppose f is known up to a multiplicative factor, i.e

$$f(x) = h(x)/C,$$

where $C > 0$ is a constant that is possibly intractable. If $q(\cdot \mid \cdot)$ satisfies the following *detailed balance* condition,

$$h(x)q(y \mid x) = h(y)q(x \mid y) \text{ for all } x \neq y \in \mathscr{X},$$

then f is a stationary distribution of (X_t).

Proof.

From the condition, $f(x)q(y \mid x) = f(y)q(x \mid y)$ for all x and $y \in \mathscr{X}$. Then

$$\int f(x)q(y \mid x)\,\mathrm{d}x = \int f(y)q(x \mid y)\,\mathrm{d}x = f(y)\int q(x \mid y)\,\mathrm{d}x = f(y)$$

where the last equality is due to the fact that given y, $q(x \mid y)$ is a probability density. □

Metropolis-Hastings (MH) algorithm

Detailed balance is the guiding principle for many MCMC algorithms. However, it does not say how to construct a kernel. One answer is provided by the well-known MH algorithm. It uses a *proposal distribution* $k(y \mid x)$ to implicitly construct a transition kernel satisfying the detailed balance condition. The construction has some similarity to rejection sampling.

MH Algorithm

Set an arbitrary x_0 with $h(x_0) > 0$. Then, for $t \geq 0$, given x_t, sample x_{t+1} as follows.

1. Draw $x^* \sim k(\cdot \mid x_t)$ and $U \sim \text{Unif}(0, 1)$.
2. Compute the *MH ratio* $R(x_t, x^*)$, where

$$R(x, y) = \frac{h(y)k(x \mid y)}{h(x)k(y \mid x)}$$

3. If $U \leq \min\{R(x_t, x^*), 1\}$, then set $x_{t+1} = x^*$, else set $x_{t+1} = x_t$.

We shall show that the MH algorithm satisfies the detailed balance condition. But first, note that $R(x_t, x^*)$ is always well-defined, because x^* can be sampled only if $f(x_t) > 0$ and $k(x^* \,|\, x) > 0$.

Since each x_{t+1} is sampled solely based on x_t, then (x_t) is a Markov chain. Let $q(y \,|\, x)$ be its transition kernel. Observe that for $y \neq x$,

$$
\begin{aligned}
h(x)q(y \,|\, x) > 0 &\iff h(x) > 0 \text{ and } q(y \,|\, x) > 0 \\
&\iff h(x) > 0,\ k(y \,|\, x) > 0,\ h(y) > 0,\ k(x \,|\, y) > 0 \\
&\iff h(y) > 0 \text{ and } q(x \,|\, y) > 0 \\
&\iff h(y)q(y \,|\, x) > 0.
\end{aligned}
$$

Therefore, the detailed balance condition is satisfied if $h(x)q(y \,|\, x) = 0$.

Next, if $h(x)q(y \mid x) > 0$, then $h(y)q(x \mid y) > 0$. In this case,

$$\begin{aligned} q(y \mid x) &= k(y \mid x)\,\text{min}\{R(x,y), 1\} \\ &= k(y \mid x)\,\text{min}\left\{\frac{h(y)k(x \mid y)}{h(x)k(y \mid x)}, 1\right\} \end{aligned}$$

Then

$$\begin{aligned} h(x)q(y \mid x) &= h(x)k(y \mid x)\,\text{min}\left\{\frac{h(y)k(x \mid y)}{h(x)k(y \mid x)}, 1\right\} \\ &= \text{min}\left\{h(y)k(x \mid y),\ h(x)k(y \mid x)\right\}. \end{aligned}$$

Likewise, since $h(y)q(x \mid y) > 0$, it is also equal to the right hand side of the equation. Therefore, the detailed balance is also satisfied for $x \neq y$ with $h(x)k(y \mid x) > 0$.

Bayesian inference

MCMC methods like the MH algorithm are particularly popular tools for Bayesian inference. Let $f(x \mid \theta)$ be a parametric family of probability densities and $p(\theta)$ be a prior distribution of parameter θ. Given data x, the posterior distribution of θ is

$$p(\theta \mid x) \propto h(\theta) := f(x \mid \theta)p(\theta).$$

A simple strategy is to use the prior $p(\theta)$ as a proposal distribution; i.e., for any θ_t, $k(\theta^* \mid \theta_t) = p(\theta^*)$. Then the MH ratio is simply the likelihood ratio

$$R(\theta_t, \theta^*) = \frac{h(\theta^*)p(\theta^*)}{h(\theta_t)p(\theta_t)} = \frac{f(x \mid \theta^*)}{f(x \mid \theta_t)}.$$

and the M-H algorithm becomes

- Draw $\theta^* \sim p$ and $U \sim$ Unif$(0,1)$. If $f(x \mid \theta^*) \geq Uf(x \mid \theta_t)$, then set $x_{t+1} = x^*$; else set $x_{t+1} = x_t$.

Gibbs sampling

Gibbs sampling is specifically adapted for multidimensional target distributions. It works by sequentially sampling from the *conditional* distributions of parts of the random vector, which are often available in closed form.

Let $\boldsymbol{X}$ be a random vector. Suppose we can partition it into random subvectors $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_p$, such that, for each $i = 1, \ldots, p$, the conditional distribution of $\boldsymbol{X}_i$ given all the other $\boldsymbol{X}_j$'s is easy to sample. With a little abuse of notation, denote by

$$f(\boldsymbol{x}_i \,|\, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_p)$$

the conditional density of $\boldsymbol{X}_i$ given $\boldsymbol{X}_j = \boldsymbol{x}_j$ for $j \neq i$.

Gibbs sampler

Set $t = 0$ and $\boldsymbol{x}^{(0)} = (\boldsymbol{x}_1^{(0)}, \dots, \boldsymbol{x}_p^{(0)})$. Then, for $t \geq 0$, given $\boldsymbol{x}^{(t)}$, sample $\boldsymbol{x}^{(t+1)}$ as follows.

1. Sample i uniformly from $\{1, \dots, p\}$.
2. Draw $\boldsymbol{x}_i^{(t+1)}$ from $f(\boldsymbol{x}_i \,|\, \boldsymbol{x}_1^{(t)}, \dots, \boldsymbol{x}_{i-1}^{(t)}, \boldsymbol{x}_{i+1}^{(t)}, \dots, \boldsymbol{x}_p^{(t)})$
3. Set $\boldsymbol{x}_j^{(t+1)} = \boldsymbol{x}_j^{(t)}$ for all $j \neq i$.

Let $q(\boldsymbol{y} \,|\, \boldsymbol{x})$ be the transition kernel of the Markov chain in the Gibbs sampler. Note that for $\boldsymbol{x} \neq \boldsymbol{y}$, $q(\boldsymbol{y} \,|\, \boldsymbol{x}) > 0$ only when there is exactly one i such that $\boldsymbol{x}_i \neq \boldsymbol{y}_i$ and for all $j \neq i$, $\boldsymbol{x}_i = \boldsymbol{y}_i$. In this case, writing $\boldsymbol{x}_{-i} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_p)$,

$$\begin{aligned} f(\boldsymbol{x})q(\boldsymbol{y} \,|\, \boldsymbol{x}) &= \frac{1}{n} f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p) f(\boldsymbol{y}_i \,|\, \boldsymbol{x}_{-i}) \\ &= \frac{1}{n} f(\boldsymbol{x}_i \,|\, \boldsymbol{x}_{-i}) f(\boldsymbol{x}_{-i}) f(\boldsymbol{y}_i \,|\, \boldsymbol{x}_{-i}) \end{aligned}$$

where the factor $1/n$ is the probability that i is sampled. Because, $\boldsymbol{y}_{-i} = \boldsymbol{x}_{-i}$,

$$\begin{aligned} f(\boldsymbol{x})q(\boldsymbol{y} \,|\, \boldsymbol{x}) &= \frac{1}{n} f(\boldsymbol{x}_i \,|\, \boldsymbol{y}_{-i}) f(\boldsymbol{y}_{-i}) f(\boldsymbol{y}_i \,|\, \boldsymbol{y}_{-i}) \\ &= \frac{1}{n} f(\boldsymbol{x}_i \,|\, \boldsymbol{y}_{-i}) f(\boldsymbol{y}) \\ &= f(\boldsymbol{y}) q(\boldsymbol{x} \,|\, \boldsymbol{y}). \end{aligned}$$

Therefore, the detailed balance condition is satisfied.

Usually, instead of randomly sampling the index i, the following variant is used. Note that for $p > 1$, the variant does not satisfy the detailed balance condition. However, under certain conditions, the Markov chain $\boldsymbol{x}^{(t)}$ generated by the sampler still has f as the stationary distribution.

Gibbs sampler with cycles
Set $t = 0$ and $\boldsymbol{x}^{(0)} = (\boldsymbol{x}_1^{(0)}, \ldots, \boldsymbol{x}_p^{(0)})$. Then, for $t \geq 0$, given $\boldsymbol{x}^{(t)}$, sample $\boldsymbol{x}^{(t+1)}$ as follows.

Draw $\boldsymbol{x}_1^{(t+1)}$ from $f(\boldsymbol{x}_1 \,|\, \boldsymbol{x}_2^{(t)}, \boldsymbol{x}_3^{(t)}, \boldsymbol{x}_4^{(t)}, \ldots, \boldsymbol{x}_p^{(t)})$.

Draw $\boldsymbol{x}_2^{(t+1)}$ from $f(\boldsymbol{x}_2 \,|\, \boldsymbol{x}_1^{(t+1)}, \boldsymbol{x}_3^{(t)}, \boldsymbol{x}_4^{(t)}, \ldots, \boldsymbol{x}_p^{(t)})$.

Draw $\boldsymbol{x}_3^{(t+1)}$ from $f(\boldsymbol{x}_3 \,|\, \boldsymbol{x}_1^{(t+1)}, \boldsymbol{x}_2^{(t+1)}, \boldsymbol{x}_4^{(t)}, \ldots, \boldsymbol{x}_p^{(t)})$.

. . . . . .

Draw $\boldsymbol{x}_{p-1}^{(t+1)}$ from $f(\boldsymbol{x}_{p-1} \,|\, \boldsymbol{x}_1^{(t+1)}, \boldsymbol{x}_2^{(t+1)}, \ldots, \boldsymbol{x}_{p-2}^{(t+1)}, \boldsymbol{x}_p^{(t)})$.

Draw $\boldsymbol{x}_p^{(t+1)}$ from $f(\boldsymbol{x}_{p-1} \,|\, \boldsymbol{x}_1^{(t+1)}, \boldsymbol{x}_2^{(t+1)}, \ldots, \boldsymbol{x}_{p-1}^{(t+1)})$.

Simulating Brownian motions

A *standard* one-dimensional (1D) Brownian motion (BM) on $[0, \infty)$ is a stochastic process $\{ W(t) : t \geq 0\}$, such that

1. $W(0) = 0$;
2. $W(t)$ is a continuous function with probability one;
3. for any $0 \leq t_0 < t_1 < \cdots < t_k < \cdots$, the increments

   $$W(t_i) - W(t_{i-1}), \quad i = 1, 2, \ldots$$

   are independent of each other; and
4. for any $0 \leq s < t$, $W(t) - W(s) \sim N(0, t - s)$.

For constants μ, $\sigma^2 > 0$, if $\sigma^{-1}[X(t) - \mu t]$ is a standard BM, then X is called a BM with drift μ and diffusion coefficient σ^2, denoted $X \sim \mathsf{BM}(\mu, \sigma^2)$. More generally, a BM with deterministic but time-varying drift $\mu(t)$ and diffusion coefficient $\sigma^2(t)$ satisfies the SDE

$$\mathrm{d}X(t) = \mu(t)\,\mathrm{d}t + \sigma(t)\,\mathrm{d}W(t),$$

i.e.

$$X(t) = X(0) + \int_0^t \mu(s)\,\mathrm{d}s + \int_0^t \sigma(s)\,\mathrm{d}W(s).$$

It has a continuous sample path almost surely and independent increments. For $0 \le s < t < \infty$, $X(t) - X(s)$ is normally distributed with

$$\mathbb{E}[X(t) - X(s)] = \int_s^t \mu(x)\,\mathrm{d}x,$$

$$\mathrm{Var}[X(t) - X(s)] = \int_s^t \sigma^2(x)\,\mathrm{d}x.$$

Random walk construction

Suppose the value of $X(0)$ is either known and fixed or can be sampled. Given $0 = t_0 < t_1 < t_2 < \ldots < t_n$,

$$D_i = X(t_{i+1}) - X(t_i), \quad i = 0, \ldots, n-1,$$

are independent and can be written as

$$D_i = \int_{t_i}^{t_{i+1}} \mu(x)\,\mathrm{d}x + \sqrt{\int_{t_i}^{t_{i+1}} \sigma^2(x)\,\mathrm{d}x}\, Z_{i+1},$$

with $Z_1, \ldots, Z_n$ iid $\sim N(0,1)$. This yields the following algorithm to sample $X(t_1)$, ..., $X(t_n)$.

1. Set or sample $X(0)$
2. Sample $Z_1, \ldots, Z_n$ iid $\sim N(0,1)$
3. For $i = 0, \ldots, n-1$, compute D_i and set $X(t_{i+1}) = X(t_i) + D_i$.
4. Return $(X(0), X(t_1), \ldots, X(t_n))$

Brownian bridge construction

For BM $W(t)$ with drift $\mu(t)$ and diffusion coefficient $\sigma^2(t)$, the random walk construction generates $W(t_i)$ from left to right. Alternatively, one may first generate the final value $W(t_n)$, then sample $W(t_{\lfloor n/2 \rfloor})$ conditional on $W(t_n)$ and progressively filling in intermediate values.

1. Useful in implementing variance reduction techniques and low-discrepancy methods;
2. Allows the "time resolution" of the sample to be increased based on already sampled $W(t_i)$;
3. The key is the conditioning formula for jointly normal random vectors.

Recall the following conditioning formula.
Let $\boldsymbol{X}_1 \in \mathbb{R}^m$, $\boldsymbol{X}_2 \in \mathbb{R}^n$ be jointly normal, such that

$$\mathbb{E}\boldsymbol{X}_i = \boldsymbol{\mu}_i, \quad \mathrm{Cov}(\boldsymbol{X}_i, \boldsymbol{X}_j) = \boldsymbol{\Sigma}_{ij}.$$

If $\boldsymbol{\Sigma}_{22}$ is of full rank, then, conditional on $\boldsymbol{X}_2 = \boldsymbol{x}$, $\boldsymbol{X}_1 \sim N(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$, where

$$\boldsymbol{\mu}_c = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_2), \quad \boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}.$$

Note that $\boldsymbol{\Sigma}_c$ is independent of $\boldsymbol{x}$.
We shall focus the Brownian bridge construction for BM$(0, 1)$. The more general case follows the same idea but a bit more complex.

To start with, given $0 \le a < t < b$, consider how to sample $W(t)$ given $W(a) = x$ and $W(b) = y$:

$$\begin{array}{ccc} a & t & b \\ W(a) = x & W(t) = ? & W(b) = y. \end{array}$$

Since $\mathrm{Cov}(W(t_1), W(t_2)) = t_1$ for any $0 \le t_1 < t_2$,

$$\begin{pmatrix} W(t) \\ W(a) \\ W(b) \end{pmatrix} \sim N\left(0, \begin{pmatrix} t & a & t \\ a & a & a \\ t & a & b \end{pmatrix}\right)$$

Take $\boldsymbol{X}_1 = W(t)$, $\boldsymbol{X}_2 = (W(a), W(b))'$. Then $\mu_1 = 0$, $\boldsymbol{\mu}_2 = \mathbf{0}$, and

$$\boldsymbol{\Sigma}_{11} = t, \quad \boldsymbol{\Sigma}_{22} = \begin{pmatrix} a & a \\ a & b \end{pmatrix}, \quad \boldsymbol{\Sigma}_{12} = (a, t).$$

Then, conditioning on $W(a) = x$, $W(b) = y$, $W(t) \sim N(\mu, \sigma^2)$, with

$$\begin{aligned}\mu = \mu_1 + \mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_2) &= (a, t)\begin{pmatrix} a & a \\ a & b \end{pmatrix}^{-1}\begin{pmatrix} x \\ y \end{pmatrix} \\ &= \frac{(b-t)x + (t-a)y}{b-a},\end{aligned}$$

and

$$\sigma^2 = \mathbf{\Sigma}_{11} - \mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1}\mathbf{\Sigma}_{21} = t - (a, t)\begin{pmatrix} a & a \\ a & b \end{pmatrix}^{-1}\begin{pmatrix} a \\ t \end{pmatrix} = \frac{(t-a)(b-t)}{b-a}.$$

Note the conditional mean of $W(t)$ is the linear interpolation of x and y.

More generally, suppose we have $W(s_i)$, for $0 = s_0 < s_1 < \ldots < s_n$ and wish to insert more time points. There are two important facts that allow this to be done. First, if $s_i < t < s_{i+1}$, then

$$\begin{aligned} &\text{Conditional distribution of } W(t) \text{ given } \{W(s_j),\ j = 1, \ldots, n\} \\ &= \text{Conditional distribution of } W(t) \text{ given } \{W(s_i),\ W(s_{i+1})\}. \end{aligned}$$

Second, for each interval (s_{i-1}, s_i), if we choose exactly one t_i, then $W(t_1)$, . . . , $W(t_n)$ are *conditionally independent*:

$$\text{given } W(s_j) = x_j,\, j = 1, \ldots, n$$
$$(W(t_1), \ldots, W(t_n)) \sim (\mu_1 + \sigma_1 Z_1, \ldots, \mu_n + \sigma_n Z_n)$$

where $Z_1, \ldots, Z_n$ are iid $\sim N(0, 1)$, and

$$\mu_i = \frac{(s_i - t_i)x_{i-1} + (t_i - s_{i-1})x_i}{s_i - s_{i-1}}, \quad \sigma_i^2 = \frac{(t_i - s_{i-1})(s_i - t_i)}{s_i - s_{i-1}}$$

For general

$$W(t) \sim \mathsf{BM}(\mu(t), \sigma^2(t))$$

one has to make several changes. First, define

$$u(t) = \int_0^t \mu(x)\,\mathrm{d}x, \quad d^2(t) = \int_0^t \sigma^2(x)\,\mathrm{d}x$$

Then for $a < t < b$, use

$$\boldsymbol{\mu}_1 = u(t), \quad \boldsymbol{\mu}_2 = \begin{pmatrix} u(a) \\ u(b) \end{pmatrix},$$

$$\boldsymbol{\Sigma}_{11} = d^2(t), \quad \boldsymbol{\Sigma}_{22} = \begin{pmatrix} d^2(a) & d^2(a) \\ d^2(a) & d^2(b) \end{pmatrix}$$

$$\boldsymbol{\Sigma}_{12} = (d^2(a), d^2(t)).$$

in the conditioning formula.

Geometric BM: one dimension, fixed drift

If ln $S(t)$ is a BM, then $S(t)$ is called a *geometric BM*. It is often specified by an SDE. For example, if S satisfies the SDE

$$\frac{\mathrm{d}S(t)}{S(t)} = \mu\,\mathrm{d}t + \sigma\,\mathrm{d}W(t),$$

then we denote $S \sim$ GBM(μ, σ^2). To solve the SDE, we may guess

$$S(t) = S(0)e^{at+bW(t)} = f(t, W(t)),$$

where $f(t,x) = S(0)e^{at+bx}$. Then by Itô's formula,

$$\begin{aligned}\mathrm{d}S(t) &= f'_t\,\mathrm{d}t + f'_x\,\mathrm{d}W(t) + \frac{1}{2}f''_{xx}(\mathrm{d}W(t))^2|_{x=W(t)}\\ &= aS(t)\,\mathrm{d}t + bS(t)\,\mathrm{d}W(t) + \frac{b^2}{2}S(t)\,\mathrm{d}t.\end{aligned}$$

Comparing the SDEs, $b = \sigma$, $a = \mu - \sigma^2/2$, and so

$$S(t) = S(0)e^{(\mu-\sigma^2/2)t+\sigma W(t)}.$$

Gaussian short rate models

The instantaneous continuously compounded short rate at time t, denoted by $r(t)$, is important. First, it determines the value $\beta(t)$ of a risk-free asset at time t by

$$\beta(t) = \beta(0)\exp\left(\int_0^t r(u)\,\mathrm{d}u\right).$$

Second, letting

$$B(t,T) = \text{time-}t \text{ price of a bond paying 1 at } T,$$

where $T > 0$ is fixed and $t \in [0,T]$, from the formula of $\beta(t)$, it is seen

$$B(0,T) = \mathbb{E}\left[\exp\left(-\int_0^T r(u)\,\mathrm{d}u\right)\right], \quad B(t,T) = \mathbb{E}\left[\exp\left(-\int_t^T r(u)\,\mathrm{d}u\right)\right.$$

and the instantaneous continuously compounded forward rate fixed at t for $T > t$ is

$$f(t,T) = -\frac{\partial}{\partial T}\ln B(t,T).$$

It is thus important to model $r(t)$. Two simple but important models are as follows.
Vasicek model is

$$\mathrm{d}r(t) = \alpha(b(t) - r(t))\,\mathrm{d}t + \sigma\,\mathrm{d}W(t)$$

1. α and σ: positive deterministic constants
2. $b(t) > 0$: deterministic, specified according to bond prices

Under Vasicek model $r(t)$ is pulled toward $b(t)$ at speed $\alpha > 0$. Long-run interest rate can be modeled by setting $b(t) \equiv b$. The result is the Ornstein-Uhlenbeck process
Continuous-time Ho-Lee model is

$$\mathrm{d}r(t) = g(t)\,\mathrm{d}t + \sigma\,\mathrm{d}W(t).$$

In practice, $g(t)$ may be specified according to observed bond prices.

Both models are special cases to the *Gaussian Markov process*

$$\mathrm{d}r(t) = [g(t) + h(t)r(t)]\,\mathrm{d}t + \sigma(t)\,\mathrm{d}W(t)$$

where g, h and σ are deterministic functions of t. Let

$$H(t) = \int_0^t h(s)\,\mathrm{d}s.$$

By Itô's formula

$$\mathrm{d}(e^{-H(t)}r(t)) = e^{-H(t)}g(t)\,\mathrm{d}t + e^{-H(t)}\sigma(t)\,\mathrm{d}W(t),$$

and so the general solution is

$$r(t) = e^{H(t)}r(0) + \int_0^t e^{H(t)-H(s)}g(s)\,\mathrm{d}s + \int_0^t e^{H(t)-H(s)}\sigma(s)\,\mathrm{d}W(s)$$

In particular, Vasicek model has solution

$$r(t) = e^{-\alpha t} r(0) + \alpha \int_0^t e^{-\alpha(t-s)} b(s)\, \mathrm{d}s + \sigma \int_0^t e^{-\alpha(t-s)}\, \mathrm{d}W(s)$$

and Ho-Lee model has solution

$$r(t) = r(0) + \int_0^t g(s)\, \mathrm{d}s + \sigma W(t).$$

To simulate the models, first consider the general process. Its simulation is based on the following Markov property: conditional on $r(\tau)$, $\tau \in [0, u]$, for $t > u$,

$$r(t) \sim N\left(e^{H(t)-H(u)} r(u) + \mu(u, t),\, \sigma_r^2(u, t)\right),$$

with

$$\mu(u, t) = \int_u^t e^{H(t)-H(s)} g(s)\, \mathrm{d}s, \quad \sigma_r^2(u, t) = \int_u^t e^{2[H(t)-H(s)]} \sigma^2(s)\, \mathrm{d}s.$$

This can be seen by using

$$e^{-H(t)} r(t) = e^{-H(u)} r(u) + \int_u^t e^{-H(s)} g(s)\, \mathrm{d}s + \int_u^t e^{-H(s)} \sigma(s)\, \mathrm{d}W(s).$$

This result implies that to sample $r(t)$ and $0 = t_0 < t_1 < t_2 < \cdots$, one can do the following.

1. Sample $Z_1, Z_2, \ldots$ iid $\sim N(0, 1)$.
2. For $i = 0, 1, \ldots$, set

$$r(t_{i+1}) = e^{H(t_{i+1})-H(t_i)} r(t_i) + \mu(t_i,\, t_{i+1}) + \sigma_r(t_i,\, t_{i+1}) Z_{i+1}$$

The above procedure can be applied to Vasicek model and Ho-Lee model easily. Just note that in Vasicek model, $H(t) = -\alpha t$ and

$$\mu(u,\, t) = \alpha \int_u^t e^{-\alpha(t-s)} b(s)\,\mathrm{d}s, \quad \sigma_r^2(u, t) = \frac{\sigma^2}{2\alpha}(1 - e^{-2\alpha(t-u)}).$$

while in Ho-Lee model, $H(t) = 0$ and

$$\mu(u,\, t) = \int_u^t g(s)\,\mathrm{d}s, \quad \sigma_r^2(u, t) = (t - u)\sigma^2.$$

We need to know $b(s)$ and $g(s)$ in order to simulate the models. The question here is whether they can be set according to observations. In particular, since the forward rates $f(0, t)$ are observed, it would be nice if b and g can be set by it.

It turns out this indeed is the case. To get formulas for b and g in terms of $f(0, t)$, for the general case, $B(0, T) = \mathbb{E}[e^{-R(T)}]$, with $R(T) = \int_0^T r(u)\,\mathrm{d}u \sim$ Normal. Then

$$B(0, T) = \exp\left(-\mathbb{E}(R(T)) + \tfrac{1}{2}\mathrm{Var}(R(T))\right).$$

However,

$$B(0, T) = \exp\left(-\int_0^T f(0, t)\,\mathrm{d}t\right).$$

Therefore

$$\mathbb{E}(R(T)) - \tfrac{1}{2}\mathrm{Var}(R(T)) = \int_0^T f(0, t)\,\mathrm{d}t.$$

This relation can be used to set $b(s)$ or $g(s)$ based on observed $f(0, t)$.

For Ho-Lee model, this is particularly nice. One get

$$g(t) = \frac{\partial f(0,t)}{\partial t} + \sigma^2 t.$$

Then, in the simulation, one can use

$$\begin{aligned} r(t_{i+1}) &= r(t_i) + \int_{t_i}^{t_{i+1}} g(s)\,\mathrm{d}s + \sigma\sqrt{t_{i+1} - t_i} Z_{i+1} \\ &= r(t_i) + [f(0,t_{i+1}) - f(0,t_i)] + \tfrac{1}{2}\sigma^2[t_{i+1}^2 - t_i^2] + \sigma\sqrt{t_{i+1} - t_i} Z_i. \end{aligned}$$

In the iteration, only $f(0,t_i)$ are needed (and can be observed), and there is no need to differentiate $f(0,t)$.

For Vasicek model,

$$r(t_{i+1}) = e^{-\alpha(t_{i+1}-t_i)} r(t_i) + \mu(t_i, t_{i+1}) + \sigma \sqrt{\frac{1 - e^{-2\alpha(t_{i+1}-t_i)}}{2\alpha}} Z_{i+1}.$$

It turns out

$$\mu(u, t) = [f(0, t) + I(0, t)] - e^{-\alpha(t-u)}[f(0, u) + I(0, u)].$$

where

$$I(u, t) = \frac{\sigma^2}{2\alpha^2}(1 - e^{-\alpha(t-u)})^2$$

is computable. The above iteration thus is determined by $f(0, t)$, which is observable.

Square-root diffusions

The process satisfies the SDE

$$\mathrm{d}r(t) = \alpha(b - r(t))\,\mathrm{d}t + \sigma\sqrt{r(t)}\,\mathrm{d}W(t).$$

where $W \sim \mathrm{BM}(0,1)$, and $\alpha,\ b > 0$. It was proposed by Cox, Ingersoll and Ross (CIR) as a model of the short rate. In the model, $r(t)$ is pulled towards b at rate α, and as $r(t) \downarrow 0$, $\sigma\sqrt{r(t)} \to 0$, results in $r(t)$ always being positive.

The model also appears in volatility models. For example

$$\frac{\mathrm{d}S(t)}{S(t)} = \mu\,\mathrm{d}t + \sqrt{V(t)}\,\mathrm{d}W_1(t)$$

$$\mathrm{d}V(t) = \alpha(b - V(t))\,\mathrm{d}t + \sigma\sqrt{V(t)}\,\mathrm{d}W_2(t),$$

with $W_1,\ W_2$ iid $\sim \mathrm{BM}(0,1)$.

Euler discretization can be used to simulate the model *approximately*

$$r(t_{i+1}) \approx r(t_i) + \alpha(b - r(t_i))(t_{i+1} - t_i) + \sigma\sqrt{r(t_i)^+(t_{i+1} - t_i)}\, Z_{i+1}$$

with $Z_1, Z_2, \ldots$ iid $\sim N(0,1)$.
For *exact* simulation, note that $r(t)$ is Markov. Conditional on $r(t)$,

$$\frac{r(t+\Delta t)}{C(\Delta t)} \sim \chi^2_d\left(\frac{e^{-\alpha\Delta t} r(t)}{C(\Delta t)}\right), \quad \Delta t > 0,$$

where

$$C(\Delta t) = \frac{\sigma^2(1 - e^{-\alpha\Delta t})}{4\alpha}, \quad d = \frac{4\alpha b}{\sigma^2},$$

and $\chi^2_d(\lambda)$ is the non-central Gamma distribution with degree of freedom d and noncentrality parameter $\lambda \geq 0$.

The distribution $\chi_d^2(\lambda)$ is rather complicated. However, it can be characterized as follows. For $d < 1$,

$$\chi_d^2(\lambda) = \chi_{d+2N}^2, \ N \sim \text{Poisson}(\lambda/2);$$

while for $d > 1$,

$$\chi_d^2(\lambda) = (Z + \sqrt{\lambda})^2 + \chi_{d-1}^2, \ Z \sim N(0,1).$$

Simulation of CIR on $0 = t_0 < t_1 < \ldots < t_n$

```
Case 1: $d > 1$
for $i = 0, \ldots, n-1$
    set $c = C(t_{i+1} - t_i)$, $\lambda = e^{-\alpha(t_{i+1}-t_i)} r(t_i)/c$
    sample $Z \sim N(0,1)$, $X \sim \chi^2_{d-1}$
    set $r(t_{i+1}) = c[(Z + \sqrt{\lambda})^2 + X]$

Case 2: $d < 1$
for $i = 0, \ldots, n-1$
    set $c = C(t_{i+1} - t_i)$, $\lambda = e^{-\alpha(t_{i+1}-t_i)} r(t_i)/c$
    sample $N \sim$ Poisson$(\lambda/2)$, $X \sim \chi^2_{d+2N}$
    set $r(t_{i+1}) = cX$
```

In the above CIR model, α, b and σ constants. In principle, they can be made deterministic but time-varying. In particular,

$$\mathrm{d}r(t) = \alpha(b(t) - r(t))\,\mathrm{d}t + \sigma\sqrt{r(t)}\,\mathrm{d}W(t)$$

is frequently used to make the bond price function

$$t \to \mathbb{E}\left[\exp\left(-\int_0^t r(u)\,\mathrm{d}u\right)\right]$$

match a set of observed bond prices $B(0,t)$. In this case, Euler discretization can be used to sample $r(t_i)$ approximately,

$$r(t_{i+1}) \approx r(t_i) + \alpha(b(t_i) - r(t_i))(t_{i+1} - t_i) + \sigma\sqrt{r(t_i)^+(t_{i+1} - t_i)}\,Z_{i+1}$$

with $Z_1, Z_2, \ldots$ iid $\sim N(0,1)$.

The CIR can be extended in many ways. A simple multifacotr extension is to regard $r(t)$ as the sum of many factors,

$$r(t) = \sum_{i=1}^{n} X_i(t),$$

where X_i are independent square-root diffusion processes

$$\mathrm{d}X_i(t) = \alpha_i(b_i - X_i(t))\,\mathrm{d}t + \sigma_i\sqrt{r_i(t)}\,\mathrm{d}\,W_i(t)$$

with $W_1(t), \ldots, W_n(t)$ iid $\sim$ BM$(0, 1)$.

Finally, we show that the single factor CIR can be regarded as the sum of squares of independent Brownian motions. Specially, let $X_1(t)$, ..., $X_n(t)$ be independent Ornstein-Uhlenbeck processes

$$\mathrm{d}X_i(t) = -\frac{\alpha}{2} X_i(t)\,\mathrm{d}t + \frac{\sigma}{2}\,\mathrm{d}\,W_i(t)$$

where α, σ are constants and $W_1(t)$, ..., $W_n(t)$ are iid $\sim$ BM$(0,1)$, then

$$Y(t) = \sum_{i=1}^{n} X_i^2(t)$$

is a square-root diffusion

$$\mathrm{d}\,Y(t) = \alpha\left(\frac{\sigma^2 n}{4\alpha} - Y(t)\right)\mathrm{d}t + \sigma\sqrt{Y(t)}\,\mathrm{d}\,\tilde{W}(t),$$

where $\tilde{W}(t) \sim$ BM$(0,1)$ is defined via

$$\mathrm{d}\,\tilde{W}(t) = \sum_{i=1}^{n} \frac{X_i(t)}{\sqrt{Y(t)}}\,\mathrm{d}\,W_i(t).$$

Because of the above result, for n integer-valued, one can use the following alternative to simulate $r(t)$,

$$r(t_{i+1}) = \frac{1}{n} \sum_{j=1}^{n} \left[\sqrt{\frac{r(t_i)}{e^{\alpha \Delta t_i}}} + \sqrt{n C(\Delta t_i)} Z_{i+1,j} \right]^2$$

where $Z_{i+1,j}$ are iid $\sim N(0,1)$ and $\Delta t_i = t_{i+1} - t_i$.

Constant elasticity of variance (CEV) process

The process is defined via

$$\mathrm{d}S(t) = \mu S(t)\,\mathrm{d}t + \sigma S(t)^{\beta/2}\,\mathrm{d}W(t),$$

or

$$\frac{\mathrm{d}S(t)}{S(t)} = \mu\,\mathrm{d}t + \sigma S(t)^{(\beta-2)/2}\,\mathrm{d}W(t).$$

The model introduces dependency of the volatility on the asset price. It turns out that $X(t) = S(t)^{2-\beta}$ is a square-root diffusion:

$$\mathrm{d}X(t) = \left[\frac{\sigma^2}{2}(2-\beta)(1-\beta) + \mu(2-\beta)X(t)\right]\mathrm{d}t + \sigma(2-\beta)\sqrt{X(t)}\,\mathrm{d}W(t),$$

so one can simulate $S(t)$ by first simulating $X(t)$ using the procedure for square-root diffusions, and then taking $S(t) = X(t)^{1/(2-\beta)}$.

Processes with Jumps

Stock prices often exhibit statistical properties that cannot be captured by diffusion models. One argument is that this may be due to peculiar sudden events affecting individual assets but not the market as a whole. To take into account of such random events, stochastic processes with jumps can be used.

Merton's jump-diffusion model

Let $0 < \tau_1 < \tau_2 < \cdots$ be the random arrival times of events. Suppose that between the events, $S(t)$ is a diffusion as usual, e.g.

$$\frac{\mathrm{d}S(t)}{S(t)} = \mu\,\mathrm{d}t + \sigma \mathrm{d}\,W(t), \quad t \neq \tau_i,$$

on the other hand, the effect of the ith event on $S(t)$ is to suddenly change it by Y_i-fold

$$S(\tau_i) = S(\tau_i -)\,Y_i$$

where $S(\tau_i -) = \lim_{t \to \tau_i -} S(t)$ is the value of $S(t)$ just before the ith event.

To get a single SDE accounting for both diffusion and jumps, let

$$N(t) = (\text{Number of events by time } t) = \#\left\{i : \tau_i \le t\right\}.$$

The process $N(t)$ is identified with the process of arrival times $\{\tau_i\}$. Let

$$J(t) = \sum_{i:\tau_i \le t} (Y_i - 1) = \sum_{i=1}^{N(t)} (Y_i - 1).$$

The desired single SDE is

$$\frac{\mathrm{d}S(t)}{S(t-)} = \mu \,\mathrm{d}t + \sigma \,\mathrm{d}W(t) + \mathrm{d}J(t).$$

To see this, note $J(t)$ is a step function that only has jumps at arrivals of events,

$$\mathrm{d}J(t) = \begin{cases} 0 & t \notin \{\tau_i\}, \\ Y_i - 1 & t = \tau_i. \end{cases}$$

Between jumps, $S(t)$ is the usual diffusion, $\mathrm{d}S(t) = S(t)[\mu\,\mathrm{d}t + \sigma\,\mathrm{d}W(t)]$. As $\mathrm{d}J(t) = 0$ and S is continuous at t, one can write

$$\mathrm{d}S(t) = S(t-)[\mu\,\mathrm{d}t + \sigma\,\mathrm{d}W(t) + \mathrm{d}J(t)].$$

On the other hand, at time $t = \tau_i$,

$$\mathrm{d}S(t) = S(\tau_i) - S(\tau_i-) = S(t-)(Y_i - 1) = S(t-)\,\mathrm{d}J(t).$$

Now $\mu\,\mathrm{d}t + \sigma\,\mathrm{d}W(t)$ is negligible comparing to $\mathrm{d}J(t)$. So one can write, again,

$$\mathrm{d}S(t) = S(t-)[\mu\,\mathrm{d}t + \sigma\,\mathrm{d}W(t) + \mathrm{d}J(t)].$$

Suppose in the jump-diffusion process,

1) μ and σ are constants,
2) $W(t) \sim \text{BM}(0,1)$,
3) (τ_i, Y_i) are independent of $W(t)$; τ_i and Y_i can be dependent;
4) $Y_i > 0$, $i = 1, 2, \ldots$.

Then

$$S(t) = S(0) e^{(\mu - \sigma^2/2)t + \sigma W(t)} \prod_{i=1}^{N(t)} Y_i.$$

The process can be decomposed into two factors. One is the geometric BM

$$S(0) e^{(\mu - \sigma^2/2)t + \sigma W(t)}$$

The second one is $\prod_{i=1}^{N(t)} Y_i$, which is the compounded jump up to time t. We need to know how to simulate the second one.

Simulating jump times

The simplest model for jump times is Poisson process. A Poisson process with rate λ is defined by two properties

1) for disjoint time intervals, the happenings of events in them are independent of each other; and
2) for $I = [t, t + \Delta]$,

$$\begin{aligned} P\{\text{no event occurs in } I\} &= 1 - \lambda\Delta + o(\Delta) \\ P\{\text{one event occurs in } I\} &= \lambda\Delta \\ P\{\text{two or more events occur in } I\} &= o(\Delta) \end{aligned}$$

There are two ways to simulate a Poisson process with rate λ. The first one utilizes the fact that the “interarrival times”

$$\tau_{i+1} - \tau_i \quad \text{are i.i.d.} \quad \sim \text{Exp}(\lambda)$$

The second one utilizes the following facts

$$N(t) \sim \text{Poisson}(\lambda t), \quad \text{any } t > 0.$$

and given $N(t) = n$, $\tau_1, \ldots, \tau_n$ are the order statistics of $X_1, \ldots, X_n$ iid $\sim$ Unif$(0, t)$, i.e., $\tau_1 = X_{(1)}$, the smallest among X_i, $\tau_2 = X_{(2)}$, the second smallest, and so on.

To sample jump times in $[0, T]$, the methods are implemented as follows.

Method 1

```
set $\tau_0 = 0$
for $i = 1, 2 \ldots$
    sample $X \sim \text{Exp}(1)$ and set $\tau_i = \tau_{i-1} + \lambda X$
    if $\tau_i > T$
        return $\tau_1, \ldots, \tau_{i-1}$
```

Method 2

```
sample $N \sim \text{Poisson}(\lambda T)$
sample $U_1, \ldots, U_N$ iid $\sim \text{Unif}[0, 1]$
return $TU_{(1)}, \ldots, TU_{(N)}$
```

An inhomogeneous Poisson process with intensity function $\lambda(t)$ is a process satisfying

1. for disjoint time intervals, the happenings of events in them are independent of each other; and
2. for $I = [t, t + \Delta]$,

$$\begin{aligned} P\{\text{no event occurs in } I\} &= 1 - \lambda(t)\Delta + o(\Delta) \\ P\{\text{one event occurs in } I\} &= \lambda(t)\Delta \\ P\{\text{two or more events occur in } I\} &= o(\Delta) \end{aligned}$$

The process has the following important property. Given $T > 0$, let

$$Z = \int_0^T \lambda(t)\,\mathrm{d}t.$$

Note that $f(t) = \lambda(t)/Z$ is a probability density on $[0, T]$. Then

1. $N(T) \sim$ Poisson(Z); and
2. conditioning on $N(T) = n$, the arrival times in time interval $[0, T]$ are the order statistics of $X_1, \ldots, X_n$ i.i.d. $\sim f(t)$.

To sample arrival times in $[0, T]$, one can then do the following.

1. Sample $N \sim$ Poisson(Z).
2. Sample $X_1, \ldots, X_N$ iid $\sim f(t)$ and sort them into $X_{(1)} \le \ldots \le X_{(N)}$.
3. Return $X_{(1)}, \ldots, X_{(N)}$.

If $\lambda(t)$ is complicated so that either Z is unknown or $f(t)$ is difficult to sample from, then one can use the "thinning method", which is similar to rejection sampling. Let $\lambda_0(t)$ be a function satisfying $\lambda(t) \le \lambda_0(t)$. Suppose it is easy to sample a Poisson process with intensity function $\lambda_0(t)$.

1. Sample $\{\tau_i\}$ from a Poisson process with intensity $\lambda_0(t)$ in $[0, T]$.
2. For each i, retain τ_i with probability $\lambda(\tau_i)/\lambda_0(\tau_i)$.
3. Return the retained τ_i

Simulating a jump-diffusion process at fixed time points

Assume

1. $\tau_1, \tau_2, \dots$ form a Poisson process of intensity $\lambda(t)$
2. $Y_1, Y_2, \dots$ are iid
3. $\{\tau_1, \tau_2, \dots\}$, $\{Y_1, Y_2, \dots\}$ and $W(t)$ are mutually independent.

Recall the SDE for $S(t)$ is

$$\frac{\mathrm{d}S(t)}{S(t-)} = \mu\,\mathrm{d}t + \sigma\,\mathrm{d}W(t) + \mathrm{d}J(t),$$

with

$$J(t) = \sum_{i=1}^{N(t)} (Y_i - 1).$$

In general, if W_i are independent random variables that are also independent of a Poisson process $N(t)$, then the process

$$M(t) = \sum_{i=1}^{N(t)} W_i$$

is called a compound Poisson process.

Under the assumptions, $J(t)$ is a compound Poisson process. Let $X(t) = \ln S(t)$. Then

$$X(t) = (\mu - \tfrac{1}{2}\sigma^2)t + \sigma W(t) + \sum_{j=1}^{N(t)} \ln Y_j,$$

where

$$D(t) = \sum_{j=1}^{N(t)} \ln Y_j$$

is another compound Poisson process.

For $0 < t < u$,

$$X(u) = X(t) + (\mu - \tfrac{1}{2}\sigma^2)(u - t) + \sigma[W(u) - W(t)] + \sum_{j=N(t)+1}^{N(u)} \ln Y_j.$$

It is seen that only the total number of jumps between t and u matters in the contribution of the compound Poisson process. The actual jump times are not important. Therefore, given time points $t_1 < t_2 < \cdots$, $X(t_i)$ can be simulated as follows,

1. Sample $Z \sim N(0, 1)$
2. Sample $N \sim \text{Poisson}(\theta_i)$, where $\theta_i = \displaystyle\int_{t_i}^{t_{i+1}} \lambda(t)\,\mathrm{d}t$.
3. If $N = 0$, set $M = 0$; otherwise, sample $\ln Y_1, \ldots, \ln Y_N$ and set $M = \sum_{i=1}^{N} \ln Y_i$.
4. Set

$$X(t_{i+1}) = X(t_i) + (\mu - \tfrac{1}{2}\sigma^2)(t_{i+1} - t_i) + \sigma\sqrt{t_{i+1} - t_i}Z + M.$$

Simulating at jump times

If one is interested in what may happen at jump times, then it is necessary to simulate the point process $\tau_1, \tau_2, \ldots$. Note that, if $\tau_j < \tau_{j+1}$ are adjacent jump times, then

$$X(\tau_{j+1}) = X(\tau_j) + (\mu - \tfrac{1}{2}\sigma^2)(\tau_{j+1} - \tau_j) + \sigma[W(\tau_{j+1}) - W(\tau_j)] + \ln Y_{j+1}.$$

while

$$X(\tau_1) = (\mu - \tfrac{1}{2}\sigma^2)\tau_1 + \sigma W(\tau_1) + \ln Y_1,$$

where $W(t)$, $\{(\tau_i, Y_i)\}$ are mutually independent. Thus, once $\{(\tau_i, Y_i)\}$ are generated, $X(\tau_j)$ can be generated recursively. Note that in this case, τ_i and Y_i need not be independent.

Gamma processes

For the jump-diffusion with $J(t)$ a compound Poisson process, $X(t) = \ln S(t)$ is a process with independent increments. Processes with independent increments are called Lévy processes and have wide applications. Such processes $X(t)$, for example, can be used to model the assest prices

$$S(t) = S(0)e^{X(t)}.$$

Under some technical conditions, a Lévy process can be represented as the sum of a diffusion and a "pure-jump" process independent of the diffusion. A pure-jump process is one whose increments are made purely of jumps. If the number of jumps in every finite interval is almost surely finite, then the pure-jump process is a compound Poisson process. On the other hand, there are many pure-jump process that has infinitely many jumps between every $a < b$.

A Gamma process $X(t)$ is a Lévy process such that $X(0) = 0$ and for any $s < t$,

$$X(t) - X(s) \sim \text{Gamma}(a(t-s), \beta)$$

where (a, β) are the parameters of the process. It is seen that the process has stationary increments. From its definition, it is unclear whether such a process actually exists, and whether it has infinitely many jumps between any $a < b$. This becomes clear once the structure of the Gamma distributions are understood.

Clearly, a Gamma process is nondecreasing. One can use instead

$$X(t) = U(t) - D(t)$$

to model a process that has ups and downs, where $U(t)$ and $D(t)$ are two independent Gamma processes.

It is interesting that if $U(1)$ and $D(1)$ follow the same distribution, then $X(t)$ can be also represented as

$$X(t) = W(G(t)),$$

where W is a standard BM and G a Gamma process independent of W. Thus, X can be viewed as the result of applying a random time-change to W. Given $G(t)$, the variance of $X(t)$ is $G(t)$. Therefore, X is also named variance gamma process.

Like BM, the Gamma process can be simulated using random walk construction and bridge construction. The former is straightforward. The brideg construction is based on the following fact. For $0 < t_1 < \ldots < t_n$, conditioning on the values of $X(t_i)$, $i = 1, \ldots, n$, the Gamma processes on the intervals $(0, t_1)$, (t_1, t_2), ..., (t_{n-1}, t_n), and (t_n, ∞) are independent of each other. Assume that $X(t) \sim \text{Gamma}(at, \beta)$. Then conditioning on $X(t_{i-1}) = x$, and $X(t_i) = y$, for any $t_{i-1} < s < t_i$,

$$\frac{X(s) - x}{y - x} \sim \text{Beta}(a(s - t_1), a(t_2 - s)).$$

In other words, one can first draw $Z \sim \text{Beta}(a(s - t_1), a(t_2 - s))$ and then set

$$X(s) = (1 - Z)x + Zy.$$