UiO : Matematisk institutt

Det matematisk-naturvitenskapelige fakultet

Summary STK-4051/9051 Computational Statistics Spring 2021

Instructor: Odd Kolbjørnsen, oddkol@math.uio.no

Course structure

- Focus on methods
- Focus on implementing algorithms
  - Will mainly use R, not that efficient
  - For most methods, there exist efficient software
  - Focus on learning through implementation
- Some theory on why and how methods work

- Compulsory exercise in two parts
- Home exam on the same form as the compulsory exercise

STK 4051/9051 in one slide

- Optimization ~ Maximum likelihood
  - Continuous space (Gradient)
  - Discrete/combinatorial (Heuristics)
  - Missing/hidden variables (EM)
- Integration ~ Bayesian inference
  - Direct methods low dimensions
  - Importance weight and resampling
    - Variance reduction methods
  - Sequential Monte Carlo
  - Markov chain Monte Carlo
  - Variational Bayes
- Numerical methods within statistics

Maximum likelihood Theory

- For independent data:

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

- Maximum likelihood estimate: $\hat{\boldsymbol{\theta}}_{ML} = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$. Typically easier to work with the log-likelihood:

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log(f(\mathbf{x}_i;\boldsymbol{\theta}))$$

- For smooth likelihoods, necessary requirement:

$$\mathbf{s}(\boldsymbol{\theta}) \equiv \ell'(\boldsymbol{\theta}) = \mathbf{0}, \quad |\boldsymbol{\theta}| \text{ equations}$$ score function

$$\mathbf{J}(\boldsymbol{\theta}) \equiv -\ell''(\boldsymbol{\theta})$$ positive (definite), called observed Fisher information

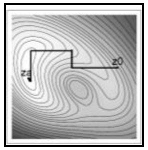
- Theory:
  - $E[\mathbf{s}(\boldsymbol{\theta})] = 0$
  - $\mathbf{I}(\boldsymbol{\theta}) \equiv -E[\ell''(\boldsymbol{\theta})] = E[\mathbf{J}(\boldsymbol{\theta})] = \text{Var}[\mathbf{s}(\boldsymbol{\theta})]$, expected Fisher information
  - For large n (and some regularity assumptions)

$$\hat{\boldsymbol{\theta}}_{ML} \approx N(\boldsymbol{\theta}, \mathbf{I}^{-1}(\hat{\boldsymbol{\theta}}_{ML})) \approx N(\boldsymbol{\theta}, \mathbf{J}^{-1}(\hat{\boldsymbol{\theta}}_{ML}))$$

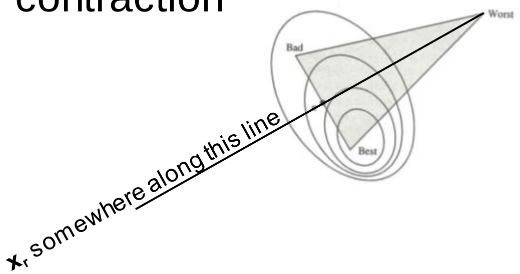
Continuous space

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \boldsymbol{B}\boldsymbol{s}\left(\boldsymbol{\theta}^{(t)}\right)$$

- Gradient based methods
  - Newton: $\boldsymbol{B} = \boldsymbol{J}\left(\boldsymbol{\theta}^{(t)}\right)^{-1}$
  - Fisher scoring, $\boldsymbol{B} = \boldsymbol{I}\left(\boldsymbol{\theta}^{(t)}\right)^{-1} = \mathrm{E}\left(\boldsymbol{J}\left(\boldsymbol{\theta}^{(t)}\right)\right)^{-1} = \mathrm{Var}\left(\boldsymbol{s}\left(\boldsymbol{\theta}^{(t)}\right)\right)^{-1}$
  - Secant, $\boldsymbol{B}$: discrete approximation of $\boldsymbol{J}\left(\boldsymbol{\theta}^{(t)}\right)^{-1}$
  - BFGS, (Quasi newton, optim in R) $\boldsymbol{B} = -\alpha\boldsymbol{M}$ [Broyden–Fletcher–Goldfarb–Shanno]
  - Ascent, $\boldsymbol{B} = \alpha\boldsymbol{I},\ \alpha > 0,$ but small enough
  - Gauss – Newton , linearize around theta, update using linear regression
- Gauss Seidel: Iterate one coordinate at the time



- Other alternatives
  - Fixed point iterations (can also be gradient based) contraction
  - Nelder – Mead (optim in R)



- Know when to stop (and why you stopped)
  - Absolute and relative error / Max iteration
  - No guarantees [except for linear equations]}

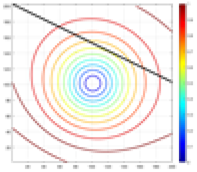
Continuous optimization «special cases»

- Iterative reweighted least squares (IRLS)

$$\boldsymbol{\beta}^{(k+1)} = \min_{\boldsymbol{\beta}} \sum w_i(\boldsymbol{\beta}^{(k)}, \boldsymbol{x}_i)\left(y_i - \boldsymbol{x}_i^T \boldsymbol{\beta}_i\right)^2$$

  - Extensively used in Generalized Linear Models

- Method of multipliers (constrained optimization)
  - $\text{minimize}_{\boldsymbol{x}} \quad \{ f(\boldsymbol{x}) \}, \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$
  - $\text{minimize}_{\boldsymbol{x},\boldsymbol{\lambda}} \left\{ f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2 + \boldsymbol{\lambda}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})\right\}$



- Alternating Direction Method of Multipliers (ADMM)

$$\begin{aligned} &\text{minimize} \quad \{ f(\boldsymbol{x}) + g(\boldsymbol{z}) \} \\ &\text{subject to} \quad \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} = \boldsymbol{c} \end{aligned}$$

For i=1 «until convergence»

1. $\boldsymbol{x}^{(i)} = \text{argmin}\left\{f(\boldsymbol{x}) + \frac{\rho}{2}\left\|\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z}^{(i-1)} - \boldsymbol{c}\right\|^2 + \boldsymbol{\lambda}^{(i-1)T}\left(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z}^{(i-1)} - \boldsymbol{c}\right)\right\}$

2. $\boldsymbol{z}^{(i)} = \text{argmin}\left\{g(\boldsymbol{z}) + \frac{\rho}{2}\left\|\boldsymbol{A}\boldsymbol{x}^{(i)} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c}\right\|^2 + \boldsymbol{\lambda}^{(i-1)T}(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c})\right\}$

3. $\boldsymbol{\lambda}^{(i)} = \boldsymbol{\lambda}^{(i-1)} + \rho\left(\boldsymbol{A}\boldsymbol{x}^{(i)} + \boldsymbol{B}\boldsymbol{z}^{(i)} - \boldsymbol{c}\right)$

  - Used for solving LASSO

Combinatorial optimization

- There are problems that are too difficult to solve exactly (NP - hard)
  - Model selection 2^p options ($p = 100 \;=>\; 1.27 \cdot 10^{30}$)
- We use heuristics when no algorithm guaranties a global maximum (within a time frame)
- Heuristics: Algorithms that find a good local optima
  - Local search
    - greedy, local optimum, use many starting points
  - Simulated annealing
    - accept proposal θ^* with probability $\min(1, \exp\{[f(\boldsymbol{\theta}^{(t)}) - f(\boldsymbol{\theta}^*)]/\tau_j\}$
    - Cooling schedule: τ_j temperature & m_j number of repeats of τ_j
  - Tabu algorithm
    - Allow downhill move when no uphill move is possible
    - Make some moves temporarily forbidden or tabu
  - Genetic algorithm- survival of the fittest
    - Use a population of solutions, paired to get next generation
    - Selection of parents/ Genetic operators / Mutations

Local neighborhood $\mathcal{N}(\boldsymbol{\theta}^{(t)})$

EM algorithm

$\boldsymbol{Y} = (\boldsymbol{X}, \boldsymbol{Z})$ complete
$\boldsymbol{X}$ observed
$\boldsymbol{Z}$ missing
Have $f_Y(\boldsymbol{y}|\boldsymbol{\theta})$

- Data are missing or "hidden", "augmented"

- If complete data, we want to maximize $\log L(\theta|Y)$
- In presence of missing data $\log L(\theta|Y)$ is unknown

Want $\max_{\theta} f_X(\boldsymbol{x}|\boldsymbol{\theta})$

$$f_X(\boldsymbol{x}|\boldsymbol{\theta}) = \int_Z f_Y(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{\theta})dz$$

$$f_X(\boldsymbol{x}|\boldsymbol{\theta}) = \frac{f_Y(\boldsymbol{y}|\boldsymbol{\theta})}{f_{Z|X}(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\theta})}$$

- We maximize:
  - The expected value of the log likelihood given observations and current estimate of parameters,

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}) = E\left[\log L(\boldsymbol{\theta}|\boldsymbol{Y}) \mid \boldsymbol{x}, \boldsymbol{\theta}^{(t)}\right] = E[\log f_Y(y|\theta)|x, \theta^{(t)}] = \int_Z \log[f_Y(y|\theta)]\, f_{Z|X}(z|x, \theta^t)dz$$

- Algorithm:
  1. E-step: Compute $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)})$
  2. M-step: Maximize $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)})$ wrt $\boldsymbol{\theta}$ to obtain $\boldsymbol{\theta}^{(t+1)}$.
  3. Return to E-step unless a stopping criterion has been met

EM Algorithm

- Mixture Gaussian clustering/ Hidden Markov Model
- EM in exponential family
  - Compute the conditional expectation of the sufficient statistics given the observed data under current estimate
  - Find the parameter value which matches the unconditional expectation of the complete data to this value

$\boxed{\boldsymbol{s}(\boldsymbol{y}) \text{ is a sufficient statistic:}}$

E-step $\boldsymbol{s}^{(t)} = E[\boldsymbol{s}(\boldsymbol{Y})|\boldsymbol{x};\theta^{(t)}]$

M-step $\theta^{(t+1)}$ solves $E[\boldsymbol{s}(\boldsymbol{Y})|\theta] = \boldsymbol{s}^{(t)}$

- Uncertainty
  - Bootstrapping
  - Numerical Differentiation
  - Empirical information $\boxed{\mathbf{I}(\theta) = \mathrm{var}[\boldsymbol{\ell}'(\theta|\mathbf{X})]}$
    - compute this as the variance of the score functions
  - Missing information $J_X(\theta) = J_Y(\theta) - J_{Z|X}(\theta)$

Observed information Complete information Missing information

Stochastic gradient algorithm

1. Gradient descent/ascent for optimizing $\ell(\boldsymbol{\theta})$:

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \alpha\ell'(\boldsymbol{\theta}^t)$$

2. $\ell'(\boldsymbol{\theta})$ may be costly to evaluate
3. $\hat{\ell}'(\boldsymbol{\theta})$ easier (e.g subsample of data)
4. Stochastic gradient algorithm

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \alpha_t\hat{\ell}'(\boldsymbol{\theta}^t)$$

5. Convergence results if

$$\sum_t \alpha_t = \infty, \quad \sum_t \alpha_t^2 < \infty$$

Stochastic gradient decent

$$\boxed{\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \alpha^{(t)} \boldsymbol{M}^{-1} \boldsymbol{Z}\left(\boldsymbol{\theta}^{(t)}, \boldsymbol{\phi}^{(t)}\right)} \qquad \boldsymbol{Z}\left(\boldsymbol{\theta}^{(t)}, \boldsymbol{\phi}^{(t)}\right) \approx \boldsymbol{g}\left(\boldsymbol{\theta}^{(t)}\right)$$

Stochastic element gradient

- Requirements on the sequence $\{\alpha_t\}$:

$$\alpha_t > 0 \tag{A-1}$$

$$\sum_{t=2}^{\infty} \frac{\alpha_t}{\alpha_1 + \cdots + \alpha_{t-1}} = \infty \tag{A-2}$$

$$\sum_{t=1}^{\infty} \alpha_t^2 < \infty \tag{A-3}$$

Note that (A-2) implies $\sum_{t=1}^{\infty} \alpha_t = \infty$

- Requirements on the function $g(z)$ combined with its estimate:

$$\exists \delta \geq 0 \text{ such that } g(x) \leq -\delta \text{ for } x < \theta^* \text{ and } g(x) \geq \delta \text{ for } x > \theta^*. \tag{A-4}$$

$$E[Z(\theta; \phi)] = g(\theta) \text{ and } \Pr(|Z(\theta; \phi)| < C) = 1 \tag{A-5}$$

Stochastic gradient decent

- Spatial data
- Neural nets

$$R(\theta) = \sum_{i=1}^{N} R_i(\theta)$$

$$R_i(\theta) = \left(y_i - f(x_i)\right)^2$$

$$f(X) = \sum_{m=1}^{M_{NN}} \beta_m \sigma(\alpha_m^T X + \alpha_0)$$

  - At top level. compute:

$$\delta_i = -2\left(y_i - f(x_i)\right), \qquad \forall i$$

  - At hidden level, compute

$$s_{m,i} = \sigma'(\alpha_m^T x_i)\beta_m \delta_i, \qquad \forall (i,m)$$

  - Evaluate:

$$\frac{\partial R_i(\theta)}{\partial \beta_m} = \delta_i z_{m,i} \;\; \& \;\; \frac{\partial R_i(\theta)}{\partial \alpha_{m,l}} = s_{m,i} x_{i,l}$$

  - Update:

$$\beta_m^{(r+1)} = \beta_m^{(r)} - \gamma_r \sum_{i=1}^{N} \left.\frac{\partial R_i}{\partial \beta_m}\right|_{\theta=\theta^{(r)}}$$

$$\alpha_{m,l}^{(r+1)} = \alpha_{m,l}^{(r)} - \gamma_r \sum_{i=1}^{N} \left.\frac{\partial R_i}{\partial \alpha_{m,l}}\right|_{\theta=\theta^{(r)}}$$

Bayesian approach

- Likelihood $f(\mathbf{y}|\theta)$
- Introduce a prior $p(\theta)$ describing knowledge about θ prior to data
- Bayes theorem:

$$f(\theta|\mathbf{y}) = \frac{f(\theta)f(\mathbf{y}|\theta)}{f(\mathbf{y})}$$

$$f(\mathbf{y}) = \int_\theta f(\theta)f(\mathbf{y}|\theta)d\theta$$

- Bayesian paradigm: All relevant information about θ is contained in the posterior distribution $p(\theta|\mathbf{y})$
  - $\hat{\theta}_{post} = E[\theta|\mathbf{y}] = \int_\theta \theta p(\theta|\mathbf{y})d\theta$
  - Credibility interval (one-dimensional): $\alpha = \Pr(a < \theta < b|\mathbf{y}) = \int_a^b \ p(\theta|\mathbf{y})d\theta$
- Posterior: Updated knowledge based on both prior and data
- Numerical aspect: Bayesian approach change optimization to integration
- Many other integration problems both inside and outside statistics, will focus on

$$\mu = \int_\mathbf{x} h(\mathbf{x})f(\mathbf{x})d\mathbf{x}$$

- In many problems: $\mathbf{x}$ is high-dimensional

Integration and Monte Carlo method

- 1D methods for integration $O(n^{-r})$
- Monte Carlo method in higher dimensions (R^d)
  - MC: $O\left(n^{-1/2}\right)$ Provided: $\mathrm{var}(h(X)) < \infty$
  - Fubini $O\left(n^{-r/d}\right)$ Provided bound on the derivative of integrand
- Random number generator (RNG)
  - Reproducible randomness = assign seed in a PRNG

Monte Carlo method

- Aim (following notation from book):

$$\mu = E^{f(\mathbf{X})}[h(\mathbf{X})] = \begin{cases} \int_{\mathbf{x}} h(\mathbf{x})f(\mathbf{x}))d\mathbf{x} & \mathbf{x} \text{ continuous} \\ \sum_{\mathbf{x}} h(\mathbf{x})f(\mathbf{x}) & \mathbf{x} \text{ discrete} \end{cases}$$

- Main applications
  - Bayesian statistics
  - Models with hidden variables
- Monte Carlo:
  1. Simulate $\mathbf{X}_i \sim f(\mathbf{x}), i = 1, ..., n$
  2. Approximate μ by

$$\hat{\mu}_{MC} = \frac{1}{n}\sum_{i=1}^{n} h(\mathbf{x}_i)$$

- Properties:
  - Unbiased $E[\hat{\mu}_{MC}] = \mu$
  - If $X_1, ..., X_n$ are independent
    - Variance: $\text{var}[\hat{\mu}_{MC}] = \frac{1}{n}\text{var}[h(\mathbf{X})]$
    - Consistent: $\hat{\mu}_{MC} \to \mu$ as $n \to \infty$ if $\text{var}[h(\mathbf{X})] < \infty$
  - Estimate of variance:

$$\widehat{\text{var}}[\hat{\mu}_{MC}] = \frac{1}{n-1}\sum_{i=1}^{n}(h(\mathbf{x}_i) - \hat{\mu}_{MC})^2$$

- Main problem: How to simulate $\mathbf{X}_i \sim f(\cdot)$

Simulation techniques

- Exact methods
  - Inversion/transformation methods
  - Rejection sampling
- Approximate methods
  - Sampling importance resampling
  - Sequential Monte Carlo
  - Markov chain Monte Carlo (Chapter 7 and 8)
- Variance reduction methods
  - Importance sampling
  - Antithetic sampling
  - Control variates
  - Rao-blackwellization
  - Common random numbers

Simulation methods

- Low dimensions
  - Exact
    - Inversion/transformation methods
    - Rejection sampling
  - Approximate
    - Importance sampling
    - Sampling/importance resampling
- Higher dimensions (when low dimension methods fails)
  - Approximate
    - Sequential Monte Carlo (SMC) Sequential Importance Sampler (SIS)
    - Markov chain Monte Carlo (McMC)

Inversion and the transformation methods

Transformation: $\boldsymbol{X} = g(\boldsymbol{U})$
Special case : $X = F^{-1}(U)$ Inverse probability

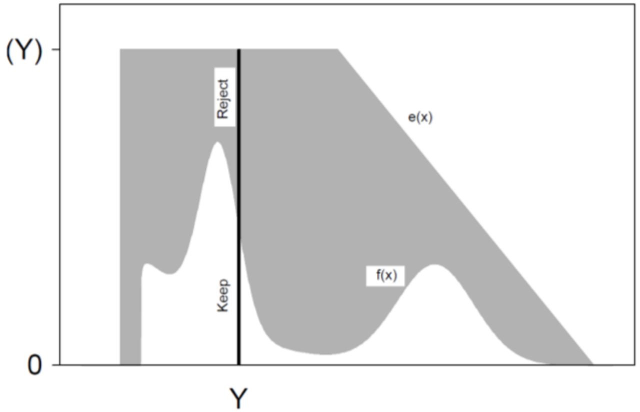
TABLE 6.1 Some methods for generating a random variable X from familiar distributions.

Distribution	Method
Uniform	See [195, 227, 383, 538, 539, 557]. For $X \sim \text{Unif}(a, b)$; draw $U \sim \text{Unif}(0, 1)$; then let $X = a + (b - a)U$.
Normal(μ, σ^2) and Lognormal(μ, σ^2)	Draw $U_1, U_2 \sim$ i.i.d. Unif(0, 1); then $X_1 = \mu + \sigma\sqrt{-2\log U_1}\cos\{2\pi U_2\}$ and $X_2 = \mu + \sigma\sqrt{-2\log U_1}\sin\{2\pi U_2\}$ are independent $N(\mu, \sigma^2)$. If $X \sim N(\mu, \sigma^2)$ then $\exp\{X\} \sim \text{Lognormal}(\mu, \sigma^2)$.
Multivariate $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Generate standard multivariate normal vector, $\mathbf{Y}$, coordinatewise; then $\mathbf{X} = \boldsymbol{\Sigma}^{-1/2}\mathbf{Y} + \boldsymbol{\mu}$.
Cauchy(α, β)	Draw $U \sim \text{Unif}(0, 1)$; then $X = \alpha + \beta\tan\{\pi(U - \frac{1}{2})\}$.
Exponential(λ)	Draw $U \sim \text{Unif}(0, 1)$; then $X = -(\log U)/\lambda$.
Poisson(λ)	Draw $U_1, U_2, \ldots \sim$ i.i.d. Unif(0, 1); then $X = j - 1$, where j is the lowest index for which $\prod_{i=1}^{j} U_i < e^{-\lambda}$.
Gamma(r, λ)	See Example 6.1, references, or for integer r, $X = -(1/\lambda)\sum_{i=1}^{r}\log U_i$ for $U_1, \ldots, U_r \sim$ i.i.d. Unif(0, 1).
Chi-square (df $= k$)	Draw $Y_1, \ldots, Y_k \sim$ i.i.d. $N(0, 1)$, then $X = \sum_{i=1}^{k} Y_i^2$; or draw $X \sim \text{Gamma}(k/2, \frac{1}{2})$.
Student's t (df $= k$) and $F_{k,m}$ distribution	Draw $Y \sim N(0, 1)$, $Z \sim \chi_k^2$, $W \sim \chi_m^2$ independently, then $X = Y/\sqrt{Z/k}$ has the t distribution and $F = (Z/k)/(W/m)$ has the F distribution.
Beta(a, b)	Draw $Y \sim \text{Gamma}(a, 1)$ and $Z \sim \text{Gamma}(b, 1)$ independently; then $X = Y/(Y + Z)$.
Bernoulli(p) and Binomial(n, p)	Draw $U \sim \text{Unif}(0, 1)$; then $X = 1_{\{U<p\}}$ is Bernoulli(p). The sum of n independent Bernoulli(p) draws has a Binomial(n, p) distribution.
Negative Binomial(r, p)	Draw $U_1, \ldots, U_r \sim$ i.i.d. Unif(0, 1); then $X = \sum_{i=1}^{r}\lfloor(\log U_i)/\log\{1-p\}\rfloor$, and $\lfloor\cdot\rfloor$ means greatest integer.
Multinomial$(1, (p_1, \ldots, p_k))$	Partition [0, 1] into k segments so the ith segment has length p_i. Draw $U \sim \text{Unif}(0, 1)$; then let X equal the index of the segment into which U falls. Tally such draws for Multinomial$(n, (p_1, \ldots, p_k))$.
Dirichlet$(\alpha_1, \ldots, \alpha_k)$	Draw independent $Y_i \sim \text{Gamma}(\alpha_i, 1)$ for $i = 1, \ldots, k$; then $\mathbf{X}^{\mathrm{T}} = \left(Y_1/\sum_{i=1}^{k} Y_i, \ldots, Y_k/\sum_{i=1}^{k} Y_i\right)$.

Rejection sampling

Easy to simulate from $g(x) \approx f(x)$.

$f(x) \leq g(x)/\alpha \equiv e(x)$ (the envelope)



Algorithm:

1. Sample $Y \sim g(\cdot)$.
2. Sample $U \sim \text{Unif}(0, 1)$.
3. If $U \leq f(Y)/e(Y)$, put $X = Y$, otherwise return to step 1

- Squeezed rejection sampling
- Adaptive rejection sampling

Want to sample from $f(x)$, but get sample from $g(x)$

- The ratio: $w(x) = f(x)/g(x)$ is important
- Rejection sampling
  - Bounding the ratio
- Importance sampling
  - Weighting with the ratio
    - Un-normalized weights $w^*(X_i) = \frac{f(\mathbf{X}_i)}{g(\mathbf{X}_i)}$ $\hat{\mu}^*_{IS} = \frac{1}{n}\sum_{i=1}^{n} h(\mathbf{X}_i)w^*(\mathbf{X}_i),$
    - Normalized weights $w(\mathbf{X}_i) = \frac{w^*(\mathbf{X}_i)}{\sum_{j=1}^{n} w^*(\mathbf{X}_j)}$ $\hat{\mu}_{IS} = \sum_{i=1}^{n} h(\mathbf{X}_i)w(\mathbf{X}_i),$
- Sampling importance Resampling (SIR)
  - Resampling with the ratio
  - Compute properties directly on resampled data
  - Proof: larger variance than importance sampling

$$\boxed{\widehat{N}_{eff} = \frac{1}{\sum_{i=1}^{n} w_i^2}}$$

Variance reduction methods

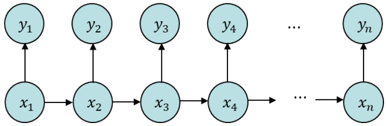
- Importance sampling
  - Normalized or un-normalized
- Antithetic sampling
  - Create two sequences with negative correlation
- Control variates
  - Use known constants for bias reduction
- Rao-Blackwellization
  - Use of conditional expectations (partially analytics)
- Common random numbers
  - Constructing pairs of high correlation

Sequential Monte Carlo

- Setting: Want to simulate from a sequence of distributions $p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})$
- Approach
  - Assume a properly weighted sample $\{(\mathbf{x}^i_{1:t-1}, w^i_{t-1}), i = 1, ..., N\}$ with respect to $p(\mathbf{x}_{1:t-1}|\mathbf{y}_{1:t-1})$
  - Use importance sampling ideas to update to samples properly weighted sample $\{(\mathbf{x}^i_{1:t}, w^i_{t-1}), i = 1, ..., N\}$ with respect to $\pi_t(\cdot)$

1. Generate $x^i_t \sim g(\cdot|\mathbf{x}^i_{1:t-1})$
2. Calculate importance weights w^i_t
3. If necessary: Resample and adjust weights

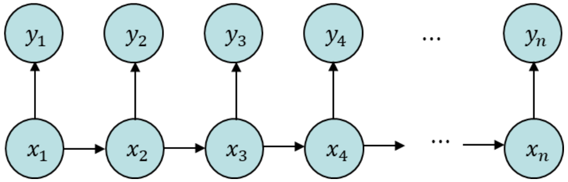
- Calculation of weights: If state space structure:
  - Markov structure on $\{x_t\}$: $p(x_t|\mathbf{x}_{1:t-1}) = p(x_t|x_{t-1})$
  - Conditional independence: $p(\mathbf{y}_{t:1}|\mathbf{x}_{1:t}) = \prod_{s=1}^t p(y_s|x_s)$
  - Markov structure on proposal: $g(x_t|\mathbf{x}_{1:t-1}) = g(x_t|x_{t-1})$



then updating of weights simplifies to

$$w^i_t = w^i_{t-1}\frac{p(x^i_t|x^i_{t-1})p(y_t|x^i_t)}{g(x^i_t|x^i_{t-1})}$$

Sequential Monte Carlo



- Origin in state space models
- Possible to use in more complex settings than state space models
  - Calculation of weights typically much more difficult
- Resampling
  - Avoid degeneracy of last point x_t
  - Will still suffer from degeneracy for x_s when $s \ll t$
- Can be extended to include parameter estimation
  - Current methods all suffer from degeneracy
  - To a variable degree

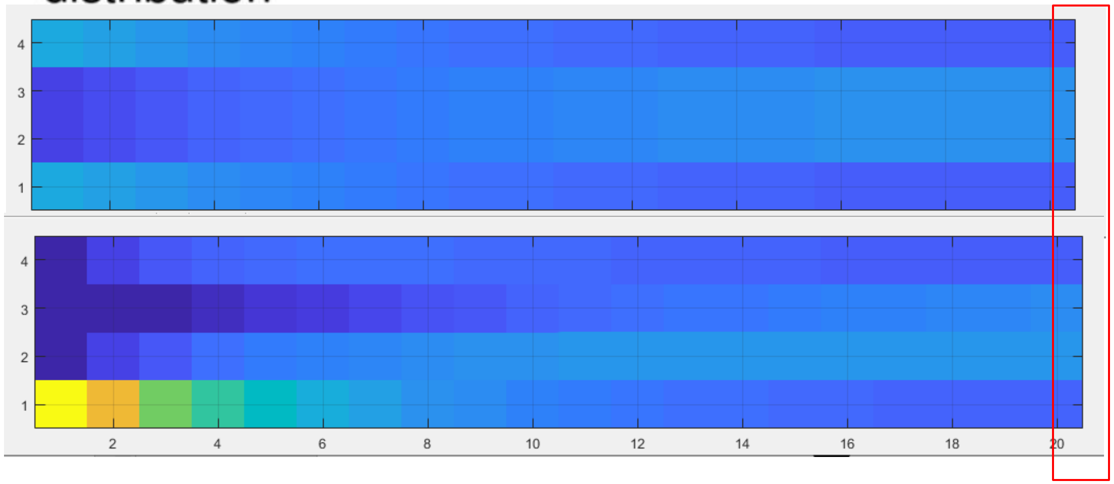
Markov chain theory general setting

- Aim: Simulate from $f(x)$
- Idea: Simulate Markov chain $\{X^{(t)}\}$ such that

$$X^{(t)} \stackrel{D}{\rightarrow} f(x)$$

$$\frac{1}{L}\sum_{t=D}^{L+D} h(X^{(t)}) \rightarrow E^f[h(X)]$$

- Markov theory: Specify $P(y|x)$ such that we have $f(x)$ as stationary distribution



Requirement for convergence

- Markov chain:
  - is Irreducible: you can visit all of parameter space
  - is Aperiodic : you do not go in loop
  - Is Recurrent : you will always return to a set
  - Has the correct stationary distribution

$$\underbrace{f(\mathbf{y}) = \int_{\mathbf{x}} f(\mathbf{x}) P(\mathbf{y}|\mathbf{x}) d\mathbf{x}}$$

Detailed balance:

$$f(y)P(x|y) = f(x)P(y|x)$$

Sufficient for stationary distribution

No guarantee for the other three

Classes of MCMC

Two main classes:

- Metropolis-Hastings
  1. Sample a candidate value $\mathbf{X}^*$ from a proposal distribution $g(\cdot|\mathbf{x})$.
  2. Compute the Metropolis-Hastings ratio
  $$R(\mathbf{x}, \mathbf{X}^*) = \frac{f(\mathbf{X}^*)g(\mathbf{x}|\mathbf{X}^*)}{f(\mathbf{x})g(\mathbf{X}^*|\mathbf{x})}$$
  3. Put
  $$\mathbf{Y} = \begin{cases} \mathbf{X}^* & \text{with probability } \min\{1, R(\mathbf{x}, \mathbf{X}^*)\} \\ \mathbf{x} & \text{otherwise} \end{cases}$$
- Gibbs sampling:
  1. Select starting values $\mathbf{x}^{(0)}$ and set $t = 0$
  2. Generate, in turn
  $$\begin{aligned} X_1^{(t+1)} &\sim f(x_1|x_2^{(t)}, x_3^{(t)}, \ldots, x_p^{(t)}) \\ X_2^{(t+1)} &\sim f(x_2|x_1^{(t+1)}, x_3^{(t)}, \ldots, x_p^{(t)}) \\ &\vdots \\ X_p^{(t+1)} &\sim f(x_p|x_1^{(t+1)}, \ldots, x_{p-1}^{(t+1)}) \end{aligned}$$
  3. Increment t and go to step 2.
- Formally, Gibbs sampler a special case of M.H, but usually considered as a separate class of algorithms

Hamiltonian Monte Carlo

- Hamiltonian MC (**?**):

$$\begin{aligned} \pi(\boldsymbol{q}) &\propto \exp(-U(\boldsymbol{q})) && \text{Distribution of interest} \\ \pi(\boldsymbol{q},\boldsymbol{p}) &\propto \exp(-U(\boldsymbol{q}) - 0.5\boldsymbol{p}^T\boldsymbol{p}) && \text{Extended distribution} \\ &= \exp(-H(\boldsymbol{q},\boldsymbol{p})) && H(\boldsymbol{q},\boldsymbol{p}) = U(\boldsymbol{q}) + 0.5\boldsymbol{p}^T\boldsymbol{p} \end{aligned}$$

- Note
  - $\boldsymbol{q}$ and $\boldsymbol{p}$ are independent
  - $\boldsymbol{p} \sim N(\mathbf{0}, \boldsymbol{I})$.
  - Usually dim($\boldsymbol{p}$)= dim($\boldsymbol{q}$)
- Algorithm ($\boldsymbol{q}$) current value
  1. Simulate $\boldsymbol{p} \sim N(\mathbf{0}, \boldsymbol{I})$
  2. Generate $(\boldsymbol{q}^*, \boldsymbol{p}^*)$ such that $H(\boldsymbol{q}^*, \boldsymbol{p}^*) \approx H(\boldsymbol{q}, \boldsymbol{p})$
  3. Accept $(\boldsymbol{q}^*, \boldsymbol{p}^*)$ by a Metropolis-Hastings step
- Main challenge: Generate $(\boldsymbol{q}^*, \boldsymbol{p}^*)$
  - Leapfrog is one possibility

Variational inference

- Bayesian inference: $p(\boldsymbol{z}|\boldsymbol{x})$
- Approximate $p(\boldsymbol{z}|\boldsymbol{x})$ by a simpler $q^*(\boldsymbol{z})$
- Perform inference by

$$E[h(\boldsymbol{z})|\boldsymbol{x}] \approx \int_{\boldsymbol{z}} h(\boldsymbol{z})q^*(\boldsymbol{z})d\boldsymbol{z} \qquad (*)$$

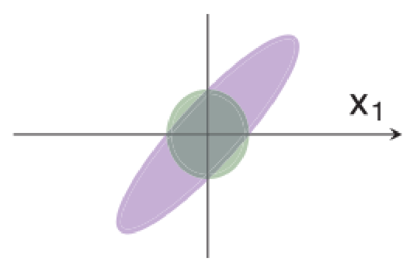
$$q^*(\boldsymbol{z}) = \underset{q(\boldsymbol{z})\in\mathcal{Q}}{\arg\min}\, \mathrm{KL}(q(\boldsymbol{z})||p(\boldsymbol{z}|\boldsymbol{x}))$$

$$\mathrm{ELBO(q)} = E^q\,(\log p(\boldsymbol{Z},\boldsymbol{x})) - E^q(\log q(\boldsymbol{z}))$$

- CAVI = Coordinate ascent variational inference

- Integration problem now mainly transformed to an optimization problem
- Mean-field approximation:

$$q(\boldsymbol{z}) = \prod_{j=1}^{m} q_j(z_j)$$



STAN

- Data
  - Real numbers with constraints
  - $\boldsymbol{y},\ \boldsymbol{\sigma}$
- Transformed data: (not a good name)
  - Real numbers and equations executed once
  - Typically fixed hyper parameters
  - alpha $= 1$, beta $= 1$
  - Any variable that is defined wholly in terms of data or transformed data should be declared and defined in the transformed data block.
- Parameter
  - The random variables we will sample
  - $\boldsymbol{\eta} = (\eta_1, \dots, \eta_p),\ \mu, \tau$
- Transformed parameters
  - $x_i = \tau \cdot \eta_i + \mu$

- Model
  - Prior: $p(\boldsymbol{\eta}, \mu, \tau)$
  - Likelihood: $p(\boldsymbol{y}|\boldsymbol{x}, \mu, \tau)$
- Generated quantities
  - $h(\boldsymbol{x}, \mu, \tau\ )$

$$E[h(\boldsymbol{x}, \mu, \sigma)|y] = \int_{\boldsymbol{z}} h(\boldsymbol{x}, \mu, \tau) p(\boldsymbol{x}, \mu, \tau|y)\, d\boldsymbol{x} d\mu d\sigma$$

"Adaptive Hamiltonian MC"

- No need to use conjugate priors
- Unlike BUGS (or other Gibbs based samplers), avoid super vauge priors if you can, i.e. `inv_gamma(0.1,0.1)`

General remarks

- (Almost) all methods discussed are iterative
- General (convergence) properties available for (almost) all methods
- Not obvious which method to use for a specific problem
  - If possible, use different methods to be sure that you have obtained the right results
- Efficiency of a particular method depend on many tuning-parameters (which are application dependent)
- Partial analytical derivations can in many cases be benificial
  - Use of gradients
  - Conditional distributions
  - Dimension reduction in optimization
  - Rao-Blackwellization in simulation

Syllabus -requirements

- Main textbook: Givens and Hoeting (2012)
  - Chapter 1 - Background Will only be referred to when needed
  - Chapter 2 - Optimization General methods, will briefly be discussed
  - Chapter 3 - Combinatorial optimization
  - Chapter 4 - The EM algorithm
  - Chapter 5 - Numerical integration General methods, will briefly be discussed
  - Chapter 6 - Monte Carlo methods
  - Chapter 7 - Markov Chain Monte Carlo
  - Chapter 8 - Advanced topics in MCMC – orientation/ as examples
  - Chapter 9 - Bootstraping
- Some additional material
  - ADMM: Alternating directions methods of moments (Slides)
  - Sequential Monte Carlo (Note)
  - Stochastic gradient methods (Note)
  - Variational inference (Slides)
  - Hamiltonian Monte Carlo / STAN (Slides)
- Example code and exercises

STK 9051

+ Article ADMM

+ Article VI

+ Article HMC

Machine learning or STK 4051/9051

- Complex models
- Algorithms for optimization
- Stochastic gradient
- Sparse coding
- Deep neural nets
- Probabilistic programming
  - Sampling
  - Variational Inference

- Large data sets
  - High performance computing
  - GPU

Course has given basic insight to important engines covering major parts of current activity

You have programmed your self to have a deeper understanding

Has not been focus, but is important in applications

=>Talk to the IT guy