Markov Chain Monte Carlo Algorithms

New Methods for Decision Making

Mark A. Powell
Mechanical Engineering
Systems Engineering
University of Idaho

Introduction

- **Topics for This Evening**
  - **Previous Seminar Synopses**
    - ***Safety* - Space Shuttle Orbital Debris Avoidance Flight Rule**
    - ***Reliability* - USCG C130 Preventative Maintenance**
    - ***Quality* - Automaker New Vehicle Production Decision**
  - **Markov Chain Monte Carlo Algorithms**
    - **Metropolis-Hastings Algorithm**
    - **Gibbs Sampling Algorithm**
  - **Integral Solution Example: Analytical vs. Ordinary Monte-Carlo vs. *MCMC***
  - **Closing and Questions**

Previous Seminars

- **Safety *Assurance* - Space Shuttle Orbital Debris Avoidance Flight Rule (Maneuver Decision)**
  - **Classical Approach Misapplied**
  - ***Arbitrary* Maneuver Decisions**
- **Reliability *Assurance* - USCG C130 Preventative Maintenance Interval Decision**
  - **Classical Approach Inconclusive**
  - ***PRA/MCMC* Solution Allowed Easy Decision and *Significant Cost Savings***
- **Quality *Assurance* - Automaker New Vehicle Production Decision**
  - **Classical Test Plan Very Expensive, Low Probability of Test Passage if Requirement Met**
  - ***PRA/MCMC* Test Plan *Cheaper*, *Easier*, *Adaptable***

Synopsis of Previous Seminars

- ***High*** **Levels of Assurance Required**
- **Non-event Data Available**
- **Decisions based on Classical Methods were all *Faulty*, if even Made**
- ***PRA*** **Methods Considered**
  - **Took advantage of Non-Event Data**
  - **All Produced Very Complex, Multivariate, Analytically Intractable and Unusable Solutions**
- ***MCMC*** **Methods Applied to *PRA* Solutions**
  - **Enabled Use of *PRA* Solutions**
  - **Provided Much Better, More Believable, and Less Costly Decisions**

A Word about R

- ***R* - A GNU version of S (or S+) – a language and environment for statistical computing**
  - **An integrated suite of software facilities for data manipulation, calculation, and graphics**
  - **Runs on Unix, Windows (9x and up) and MacOS platforms**
  - **Links with C, C++ and Fortran code**
  - **Open source, supported fully by CRAN – the Comprehensive R Archive Network**
  - **Downloadable for Free from**
    http://cran.r-project.org/
  - **Will be used for Examples in Seminar**

What is Monte Carlo Integration?

- **An Example**
  - **Suppose we model our uncertainty about *x* using a Standard Normal Model and want to know P(-2<*x*<1), the Integral of a N(0,1) Model over the region (-2, +1)**
  - **We can just Look this Up in our Tables or Use *R* or do it by Hand**
  - $$\int_{-2}^{1} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx = P\left(Z \le 1 \mid z \sim N(0,1)\right) - P\left(Z \le -2 \mid z \sim N(0,1)\right)$$

    $$= 0.8413 - 0.0228 = 0.8185$$
  -
```
> pnorm(1)-pnorm(-2)
[1] 0.8185946
```
R Code

The Monte Carlo Approach

- **We can do the Integral *Numerically* Using *Monte Carlo***
  - **Most Statistical Packages (especially including *R*) have functions to provide *Independent Random Samples* of *Named* and *Recognized* Distributions**
  - **E.g., in *R*, we can get *N* samples from a N(0,1) using the Built-in function *rnorm(N,0,1)***
  - **To Compute the Integral from *N* samples, we just *count* the number of samples *between -2 and 1* and divide by *N***
- **Here is the *R* code where *N*=5000**

```
> x<-rnorm(5000,0,1) # get the 5000 standard normal samples
> cnt<-0 # zero our counter
> for (i in 1:5000){  # Count up those inside the interval
+   if (x[[i]]>=-2) if (x[[i]]<=1) cnt<-cnt+1
+ }
> cnt/5000
[1] 0.8208
```

- **Our Analytical Result was 0.8185, error of only 0.0023 - *not Bad!***

Monte Carlo Notes

- **Monte Carlo Integration is considered a *Brute Force* Method**
- ***Very Complex* Monte Carlo Simulations Can be Produced if Considering *all* Sources of Uncertainty**
- **Marginal, Conditional, and Joint Probability Integrals Can be Performed with Monte Carlo, as Well as Mixtures**
- **However, Monte Carlo Integration *ONLY* works for *Recognizable* Models for *Independent* Sources of Uncertainty**
  - **Even for Some Very Simple Statistical Problems, our Model for the Quantity of Interest may be Unrecognizable**
  - ***PRA* Solutions are Very Often *Dependent Multivariate* and *Unrecognizable* Models**
- ***What Can We Do?***

The Answer is: Markov Chain Monte Carlo

- **Done *Properly*, Most Problems in Engineering and Science are *not Really Amenable* to *Analytical* or Ordinary *Monte Carlo Methods***
  - **Classical Solution Models are *Rarely Recognizable***
  - ***PRA* Solution Models are *Rarely Recognizable***
- **About *15* Years ago, *Markov Chain Monte Carlo* (*MCMC*) Methods were Developed**
  - **About *10-12* Years ago, the literature revealed *MCMC* was being Applied to *Statistical Problems***
  - **About *5-8* Years ago, the literature revealed *MCMC* was being Applied to *Decision Problems* in *Biological* and *Medical Sciences***
- **Even *Today*, the Literature Reveals *Very Sparse* Application of *MCMC* to Solve *Engineering* Decision *Problems***

What is MCMC?

- **First: *Any Problem* with a Nice *Analytical* and *Recognized Model* can be Solved with *MCMC Methods***
- **Second: *Any Problem Solvable* with ordinary *Monte Carlo Methods* can be solved with *MCMC Methods***
- **Third: The *Converses* of these are *NOT True***
- **With Ordinary *Monte Carlo* Methods, the *Random Samples* are *all Independent***
- **A *Markov Chain* can Also be Produced that contains Samples Over the *RV Domain* at the *Right Frequencies* (*Density*) - but Each Sample in the Markov Chain is *Related* to the Previous Sample**
- ***Who Cares?!!!* As long as the RV Domain is covered at the Right Frequencies, *Our Numerical Integral will be the Same!***

Some Remarkable Capabilities with MCMC

- **Markov Chains can be Produced that Cover the RV Domain for *ANY* Uncertainty Density Model**
  - **Named and *Recognized* Models**
  - ***Unrecognized Analytical* Models**
  - **Complex *Arbitrary* Models**
  - ***Even Improper* Models**
  - ***Even Joint* Models with more than *10,000* Conditional Random Variables (multivariate)**
- ***Marginal* Samples for Each Joint Random Variable are *Immediately* Available**
- **Full *Joint* and *Conditional* Models of *Complex Functions* (most Decision Discriminators) of the Random Variables are *Immediately* Available**

However, There is a Catch

- **Markov Chains for *MCMC* must be *Tuned* by Hand (and eye)**
  - **Need *Burn-in* for *Stationarity***
  - **Need *Adjustment* for Density *Stability***
- **The Solutions However are Quite Simple**
  - ***Ignore Early Samples* for Burn-in**
  - ***Visually Determine* if *Stability* is Achieved**
- **The Basic Algorithm Used to Generate the Markov Chains is Unbelievably Simple**
  - ***Metropolis-Hastings* Algorithm**
  - **Easy to Tune, Using Sample Acceptance Ratio and Visual Inspection of Samples**

The Metropolis-Hastings Algorithm

- **To Start, formulate the *PRA* Solution density *pd(Θ|data)*, and select a proposal *Step Size dΘ***
- **Start with any legal value: $\Theta_i = \Theta_1$**
- **Repeat This Loop to get new samples**
  - **Propose a new sample: $\Theta_{i+1} = \Theta_i + \Delta\Theta$,**
    where $\Delta\Theta \sim U(-d\Theta, d\Theta)$, a Simple *Uniform* Sample
  - **Calculate the ratio (α) of the Proposed PRA Solution Density to Previous PRA Solution Density:**
    $\alpha = pd(\Theta_{i+1}|data) / pd(\Theta_i|data)$
  - **Obtain a sample *u* from a uniform distribution: $u \sim U(0,1)$**
  - **If $u < \alpha$, then accept the proposed sample Θ_{i+1},**
    else, set the new sample to the previous one: $\Theta_{i+1} = \Theta_I$
  - **Maintain a Count of the Accepted Proposed Samples**

Some M-H Heuristics

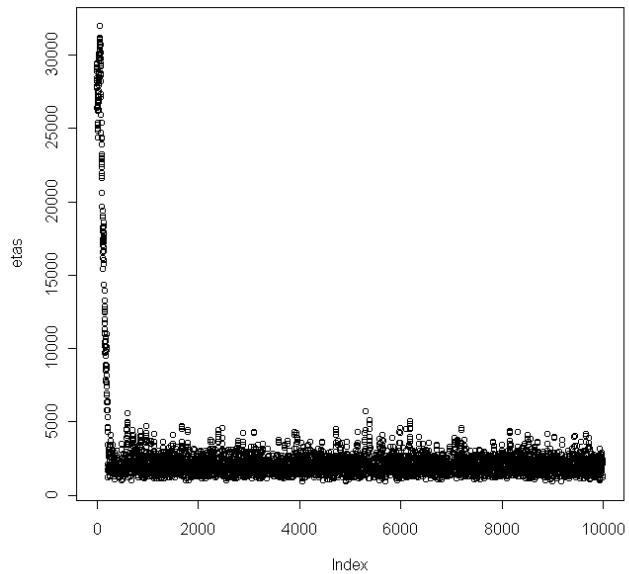
- **Use short test chains (~1,000 points) to tune**
  - **Means are good starting points for quicker *Burn-in***
  - **Two to Three Standard Deviations (2-3 σ) are good first proposal step sizes**
- **Use Logarithms of *PRA* Solution Density Functions**
- **Parameter-by-parameter (inner loop) acceptance testing is easier to *tune* than vector-at-once acceptance testing**
- **Beware**
  - **Changes on proposal step size for one parameter may change acceptance ratios for other parameters**
  - ***Improper Models* can show stationarity in short chains, then *Track-off* to non-stationarity in short chains, then back**
  - **If *Ignorance Priors* Produce *Improper PRA* Solution Models, *Pseudo-Ignorance Priors* Can Solve the *Stationarity Track-off* Problem**

Tuning the M-H Algorithm

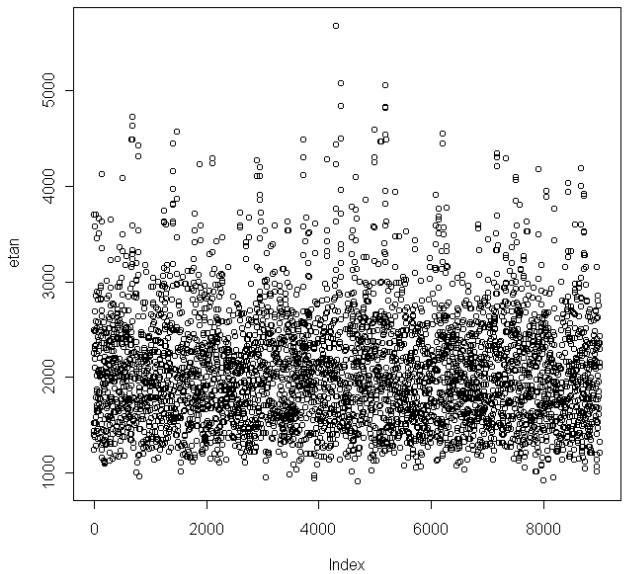
- ***Subjective Judgment*** **is usually all that is needed**
- **Need to collect data on numbers of proposed samples *accepted* (vector or on each parameter)**
- ***Acceptance Ratios*** **should be within *30-60%* range, if not, adjust the proposal step size**
  - **If ratio is too high, samples will show *tracking*, Increase step size**
  - **If ratio is too low, samples will get *stuck*, Decrease step size**
- **Must look at *Burn-In*, Convergence to a Stationary Markov Chain**

M-H Convergence to Stationarity

- **Scatterplot Marginal samples to identify point at which Markov Chain becomes stationary, use samples beyond this point – Called *Burn-In***

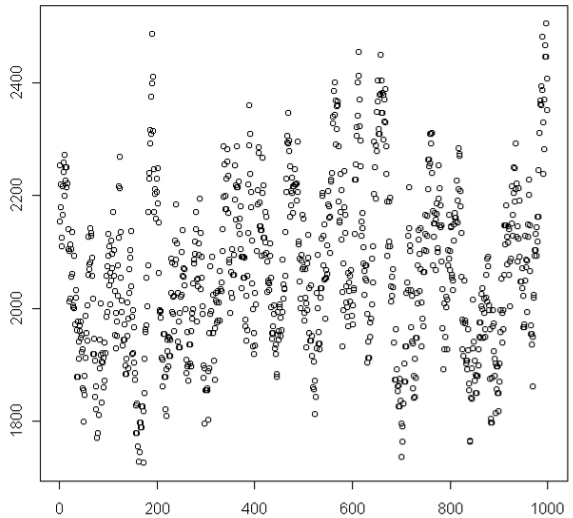


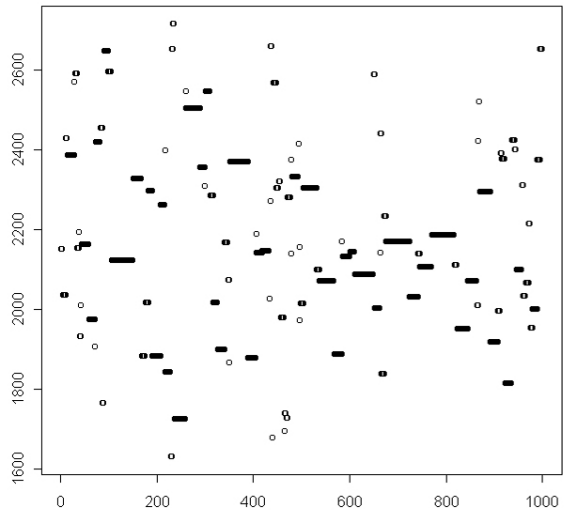
Converges before 1K samples

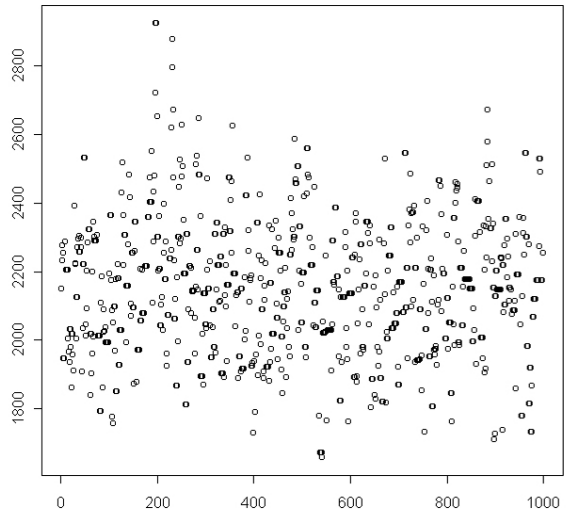


Rightmost 9,000 samples

MCMC Tuning by Acceptance Ratio and Eye







Tracking

- **Acceptance Ratio: 0.9**
- **Increase Step Size**

Stuck

- **Acceptance Ratio: 0.1**
- **Decrease Step Size**

About Right

- **Acceptance Ratio: 0.53**
- **Suitable Step Size**

Gibbs Sampling Algorithm

- **Requirement: recognizable *Conditional PRA* Solution Density Models:**

$$pd(\theta_i \mid \theta_1, \theta_2, \cdots, \theta_{i-1}, \theta_{i+1}, \theta_n, data)$$

- **Obviously, cannot have *Improper Conditional PRA* Solution Density Models**
- **Changes to M-H algorithm**
  - **Use parameter by parameter inner loop algorithm**
  - **Do not need a proposal step size $d\Theta$**
  - **To obtain a sample vector of the joint *PRA* Solution, loop for *i=1,n* proposing new sample θ_i from the its conditional Density and accept – no need to test**
  - **No need to worry about acceptance ratio or tuning**
  - ***Burn-in* still required**

Gibbs Sampling Advantages

- **Obviously, a faster algorithm than pure Metropolis-Hastings**
- **Can combine Gibbs and M-H algorithms when you know *Conditional PRA* Solution Density Models for some parameters, use standard M-H step for those you don't**

MCMC Solution to Example

- **Recall the Integral Problem**

$$\int_{-2}^{1} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx = 0.8185$$

- **Our Integrand is our Density**

$$pd(x) \propto e^{-\frac{x^2}{2}}$$

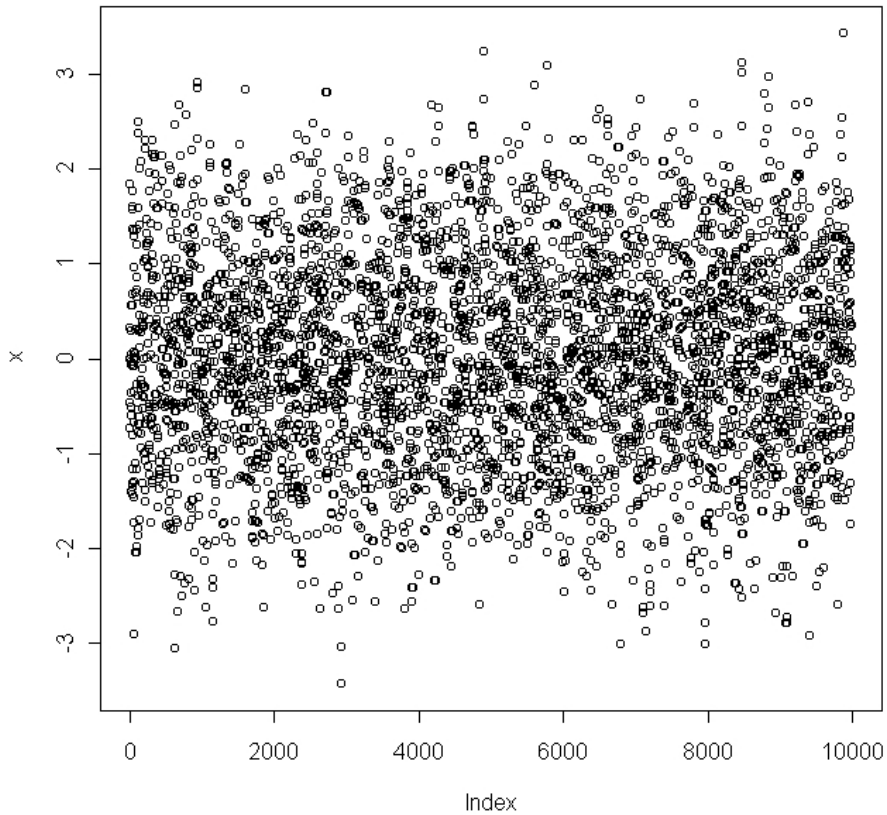
$$Log\left(pd(x)\right) \propto -\frac{x^2}{2}$$

- **Here is the MCMC Code**

```
runexample1<-function(nsamps=3000,x0=1,dx0=1) {
  #  M-H for Example 1, Initialize the Algorithm
  x<<-x0  #  Set first sample to Starting Value
  xcurr<-x0  #  Set current sample to Starting Value
  dx<-dx0
  accepts<<-0  #  Zero Acceptance Counter
  for (i in 2:nsamps) {  #  The M-H loop
    px<-xcurr+runif(1,-dx,dx)  #  Get Proposed Sample
    logalpha<--(px*px)/2+(xcurr*xcurr)/2  # Compute log(alpha)
    logu<-log(runif(1,0,1))  #  Get log uniform sample, acceptance criteria
    if (logu<logalpha) {  # Accepted proposed sample
     x<<-append(x,px)  #  Add proposed sample to vector
     xcurr<-px  #  Reset current to proposed sample
     accepts<<-accepts+1  #  Count acceptance
     }
     else { #  Did not accept proposed Sample
       x<<-append(x,xcurr)  # Add current sample to vector
     }
  }
  cat("Example 1:","\n") #  Print out some Useful Data
  cat("  Number of Samples:  ",nsamps,"\n")
  cat("  Initial x:              ",x0,"\n")
  cat("  Initial x Step:     ",x0,"\n")
  cat("  Acceptance Ratio:     ",accepts/(nsamps-1),"\n")
}
```

The MCMC Answer

```
> runexample1(nsamps=10000,x0=0,dx0=4)
Example 1:
  Number of Samples:   10000
  Initial x:              0
  Initial x Step:      4
  Acceptance Ratio:      0.3843384
> mean(x)
[1] 0.01657858
> sd(x)
[1] 0.9804565
> plot(x)
> paltxltb(x,-2,1)
[1] 0.8224
```



Error of 0.0039

Example Synopsis

- **For a Well Known Model, *MCMC* works just as well as Ordinary Monte-Carlo**
- **The M-H Algorithm was used in Developing Solutions to the Problems in All Three Previous Seminars (much more complex Problems)**
- **Slide Handouts Appended with Detailed Examples from *Probability and Statistics for Systems Engineering I* Classes**
  - **Somewhat More Complex Problems**
  - ***R* Code Provided, adaptable to Many Problems**

Potential Seminar Series Topics

- **Half or Full Day Tutorial on MCMC Algorithms and Applications**
- **Applied Decision Theory**
- **Risk Assessment and Management**
- **RAM (Reliability, Availability, and Maintainability)**
- **Flight Rule Development**
- **PRA/MCMC Applications**
- **Test Planning**
- **Others by Request**

INCOSE

- **Texas Gulf Coast Chapter Sponsorship of Seminar**
- **Contact:**
  - **Jonette Stecklein (Chapter President)**
  - **E-mail: jonette.m.stecklein@nasa.gov**

UHCL Systems Engineering Program

- **Joint Sponsorship of Seminar**
- **Contact:**
  - **Dr. Jim Helm**
  - **E-mail: helm@cl.uh.edu**

For More Information

- **Mark Powell Contact Information**
  - **Faculty Website: www.if.uidaho.edu/~powell**
  - **E-mail: powell@if.uidaho.edu**
  - **Telephone: 208-526-5641**
- **Copies of Seminar Series Charts**
  - **Published on Faculty Website**
  - **Available Post Seminar Presentation**

Questions

References

Abernethy, Robert B., *The New Weibull Handbook, Fourth Edition*. Robert B. Abernethy, North Palm Beach, Florida, 2000.

Anderson, Theodore Wilbur, *An Introduction to Multivariate Statistical Analysis, 2nd Edition*. John Wiley & Sons, Inc., New York, 1984.

Berger, James O., *Statistical Decision Theory and Bayesian Analysis, Second Edition*. Springer-Verlag, New York, 1980.

Box, George E. P., and Tiao, George C., *Bayesian Inference in Statistical Analysis*. John Wiley & Sons, Inc., New York, 1973.

Clemen, Robert T., and Reilly, Terence, *Making Hard Decisions*. Duxbury, Pacific Grove, CA, 2001.

Daniels, Jesse, Werner, Paul W., and Bahill, A. Terry, *Quantitative Methods for Tradeoff Analyses*. *Systems Engineering*, Volume 4, John Wiley & Sons, Inc., New York, 2001.

Gamerman, Dani, *Markov Chain Monte Carlo*. Chapman & Hall, London, 1997.

George, Larry, *The Bathtub Curve Doesn't Always Hold Water*; http://www.asqrd.org/articleBathtub. American Society for Quality, Reliability Division, 2002.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. J., *Markov Chain Monte Carlo in Practice*. Chapman & Hall, Boca Raton, Florida, 1996.

Hammond, John S., Keeney, Ralph L., and Raiffa, Howard, *Smart Choices, A Practical Guide to Making Better Decisions*. Harvard Business School Press, Boston, 1999.

More References

Jefferys, William H., and Berger, James O., *Ockham's Razor and Bayesian Analysis*. *American Scientist*, Volume 80, Research Triangle Park, NC, 1992.

Jeffreys, Harold, *Theory of Probability*. Oxford University Press, Oxford, 1961.

Krause, Andreas, and Olson, Melvin, *The Basics of S and S-Plus*. Springer-Verlag, New York, 2000.

Raiffa, Howard, and Schlaifer, Robert, *Applied Statistical Decision Theory*. John Wiley & Sons, Inc., New York, 1960.

Relex Software Corporation, *Relex Failure Data Analysis*; http://www.relexsoftware.com/. Relex Software Corporation, Greensburg, Pennsylvania, 2002.

Reliasoft Corporation, *Life Data Analysis Reference*. Reliasoft Publishing, Tucson, Arizona, 1997.

Schmitt, Samuel A., *Measuring Uncertainty, An Elementary Introduction to Bayesian Statistics*. Addison-Wesley Publishing Company, Inc., Phillipines, 1969.

Sivia, D. S., *Data Analysis, A Bayesian Tutorial*. Oxford University Press, Oxford, 1996.

Venables, William N., and Ripley, Brian D., *Modern Applied Statistics with S-Plus*. Springer-Verlag, New York, 1999.

Williams, David, *Probability with Martingales*. Cambridge University Press, Cambridge, 1991.

Detailed Examples from Probability and Statistics for Systems Engineering I Classes

Example #1

- **From Homework 06, Hayter Problem 6.1.6, Figure 6.12 for data**
  - **Data: Number of Calls Received at a Switchboard during One Minute Intervals**
  - **Should be Obvious that Uncertainty for this Data could be Well Modeled by a *Poisson* Model**
- **Hayter Problem 8.2.17 - For this Data, a Manager claims that the Switchboard *needs additional staff* because the *average number of calls* per minute is *at least 13***
- ***This is not what Upper Management wants to Hear!***

Classical Solution for Example #1

- **Our Null Hypothesis: $\mu \leq 13$, the Manager's Claim is *False***
- **We want to find *Evidence to Reject the Falsity of the Managers Claim* before Paying for more Staff**
- **We Can Use Hayter's *Recipe* on Pg. 390 for a *One-Sided t-Test* (Hypothesis Test)**
  - **Pick a Significance Level, say $\alpha = 0.1$**
  - **Very Easy using R**
- ***p* > α - We cannot Reject our Hypothesis that the Manager's Claim is False: *No new Staff!***

```
> # "Switchboard Calls"
> data<-c(11,12,15,12,14,17,10,13,9,12,
+ 15,17,14,8,12,15,10,15,12,14,12,9,10,
+ 11,7,10,14,13,12,14,10,13,14,16,9,11,
+ 18,15,9,15,17,15,11,11,11,16,9,14,9,10,
+ 11,11,8,10,16,10,9,14,12,10,11,13,16,
+ 11,6,14,15,18,14,13,9,11,10,10,14,13,
+ 13,14,14,11,15,9,11,13,14,15,9,10,11,9)
> n<-length(data)
> xbar<-mean(data)
> s<-sqrt(var(data))
> t<-sqrt(n)*(xbar-13)/s
> p<-1-pt(t,n-1)
> p
[1] 0.9971336
```

The PRA Solution

- **First, We have to Model our *Prior Uncertainty***
  - **We might not Believe this Manager (a known *Whiner*)**
  - **But to be *Absolutely Fair*, We can Use an *Ignorance Prior*, in fact a *Reference Prior* is best**
  - **The Ignorance (Reference) Prior for an Uncertain Poisson Rate Parameter is $pd(\lambda|H) \propto \lambda^{-1/2}$**
- **Now we Need to Formulate our Posterior**
  - **Our Likelihood:** $pd(data \mid \lambda, H) \propto \prod_{i=1}^{n} \frac{e^{-\lambda} \lambda^{datum_i}}{\Gamma(datum_i + 1)}$
  - **Our Prior:** $pd(\lambda \mid H) \propto \lambda^{-\frac{1}{2}}$
  - **Our Posterior:** $pd(\lambda \mid data, H) \propto \lambda^{-\frac{1}{2}} e^{-n\lambda} \prod_{i=1}^{n} \frac{\lambda^{datum_i}}{\Gamma(datum_i + 1)}$

Coding our Posterior for MCMC

- **First, Let's Take the Logarithm of our Posterior**

$$Log\left(pd\left(\lambda \mid data, H\right)\right) \propto -\frac{1}{2}Log\left(\lambda\right) - n\lambda + \sum_{i=1}^{n}\left[datum_i Log\left(\lambda\right) - Log\left(\Gamma\left(datum_i + 1\right)\right)\right]$$

- **Now, Let's code it in R**

```
lpd<-function(lamda,data) {  # Log of our Posterior
  n<-length(data)  # Get number of data
  sum<-0
  for (i in 1:n){  # Sum over data terms
    sum<-sum+data[i]*log(lamda)-lgamma(data[i]+1)
  }
  temp<-sum-log(lamda)/2-n*lamda  # Compute log posterior
  return(temp)
}
```

Coding the M-H Algorithm

- You are just not going to believe how *Simple* this Code Really is!

```
runexample1<-function(nsamps=3000,dat=data,lamda0=1,dlamda0=1) {
 #  M-H for Example 1, Initialize the Algorithm
 lamda<<-lamda0  #  Set first sample to Starting Value
 lamdacurr<-lamda0  #  Set current sample to Starting Value
 dlamda<-dlamda0
 accepts<<-0  #  Zero Acceptance Counter
 for (i in 2:nsamps) {  #  The M-H loop
  plamda<-lamdacurr+runif(1,-dlamda,dlamda)  #  Get Proposed Sample
  if (plamda>0) { # Got a legal proposal (lamda>0 is required)
   logalpha<-lpd(plamda,dat)-lpd(lamdacurr,dat)  # Compute log(alpha)
   logu<-log(runif(1,0,1))  #  Get log uniform sample for acceptance criteria
   if (logu<logalpha) {  # Accepted proposed sample
    lamda<<-append(lamda,plamda)  #  Add proposed sample to vector
    lamdacurr<-plamda  #  Reset current to proposed sample
    accepts<<-accepts+1  #  Count acceptance
   }
   else { #  Did not accept proposed Sample
    lamda<<-append(lamda,lamdacurr)  # Add current sample to vector
   }
  }
  else { # Got an illegal proposal, do not accept
   lamda<<-append(lamda,lamdacurr)  # Add current sample to vector
  }
 }
 cat("Example 1:","\n") #  Print out some Useful Data
 cat("  Number of Samples:  ",nsamps,"\n")
 cat("  Initial Lamda:              ",lamda0,"\n")
 cat("  Initial Lamda Step:     ",dlamda0,"\n")
 cat("  Acceptance Ratio:     ",accepts/(nsamps-1),"\n")
}
```

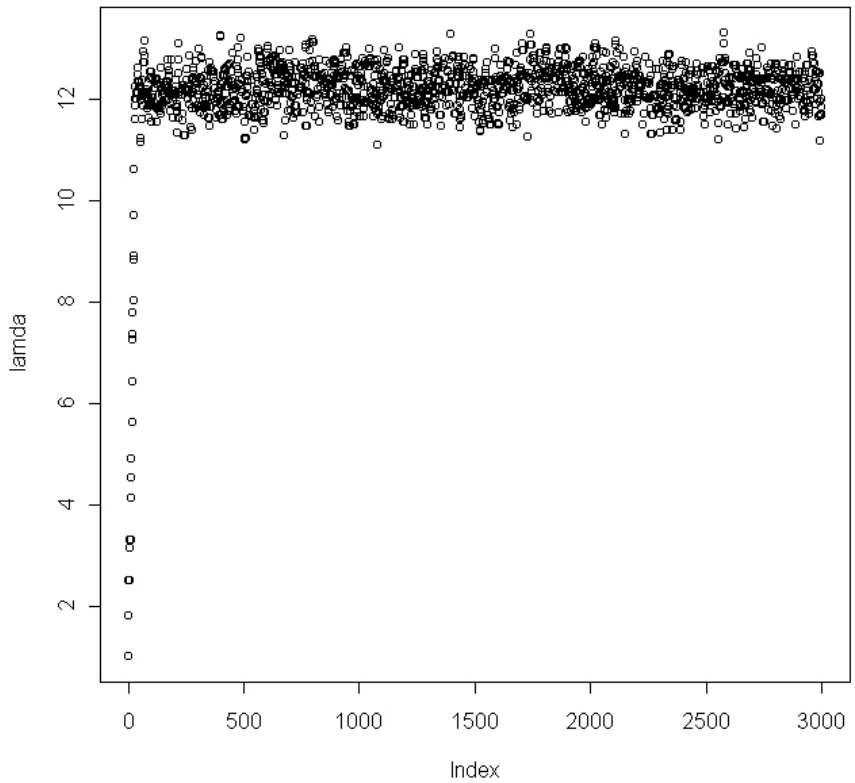
Running the M-H Algorithm

- **Copy into the R console**
  - ***MCMC Utilities* from my Faculty Website**
  - **Our Log Posterior Function**
  - **Our M-H Algorithm**
  - **Our Data (figure 6.12 data)**
- **Now *Run* the MCMC Sampler**

```
> runexample1()
Example 1:
  Number of Samples:   3000
  Initial Lamda:       1
  Initial Lamda Step:  1
  Acceptance Ratio:    0.5191731
> plot(lamda) # First Look
```

Our Initial MCMC Results

- **First, We got Lucky - our *Acceptance Ratio* is 0.51, inside our Desired Range of 30-60%**
- **Now, Lets look at our λ Samples**
- **Burn-in *Appears* to Have Occurred by Sample 100**

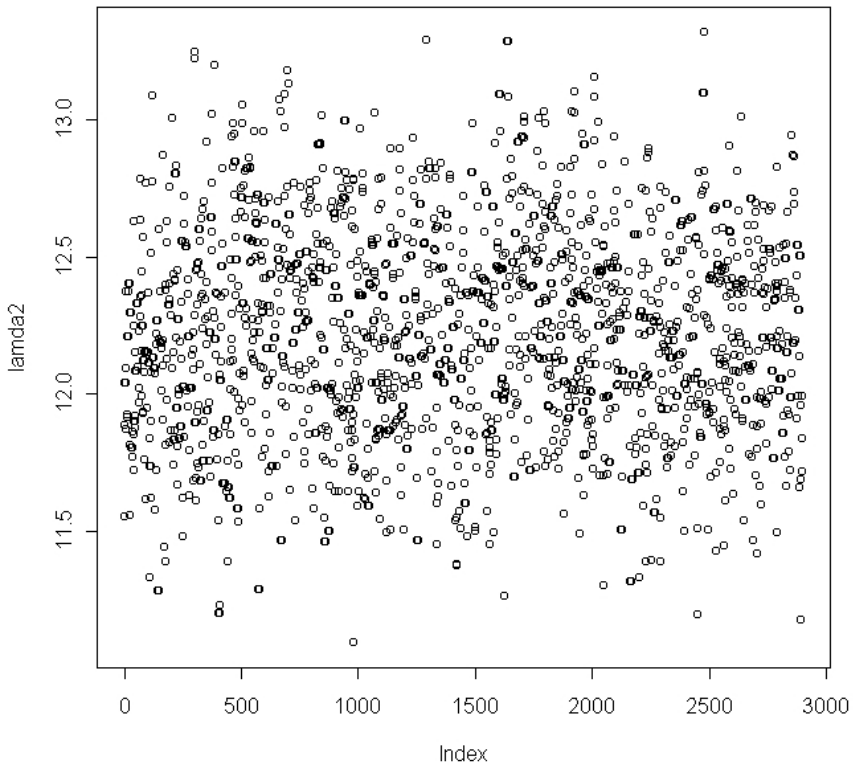


Our MCMC Samples Post Burn-in Removal

- **We removed the first 100 Samples to eliminate *Burn-in* Tracking**

```
> lamda2<-lamda[101:3000]  #Remove Burn-in
> length(lamda2)
[1] 2900
> plot(lamda2)  # Look at Post Burn-in removal
```

- **Sure Looks like *Random Sampling* Now**
  - **No Evident Tracking**
  - **No Obvious Sticking**



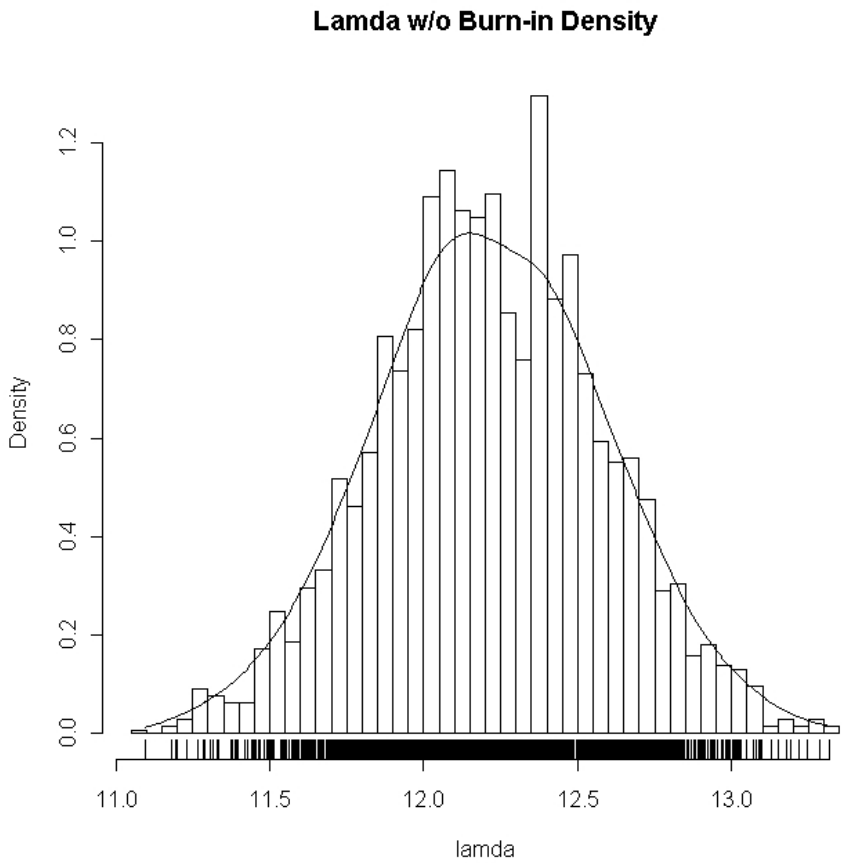
Our MCMC Posterior Uncertainty Model

- **Using the *phist* function from the Utilities on my Faculty Website**
  - **See the *Shape* of our Posterior Uncertainty**
  - **This is *Based on our Data***

```
> phist(lamda2,mainlab="Lamda w/o Burn-in Density",
+ xname="lamda",bwadj=1.8)  # Density Plot
```

- **Using the *pxgta* function from the Utilities, we find the Probability that the Manager is right is just over 1.6% given the data -**
  No new Staff!

```
> pxgta(lamda2,13)  # Probability that Manager is Right
[1] 0.01620690
```



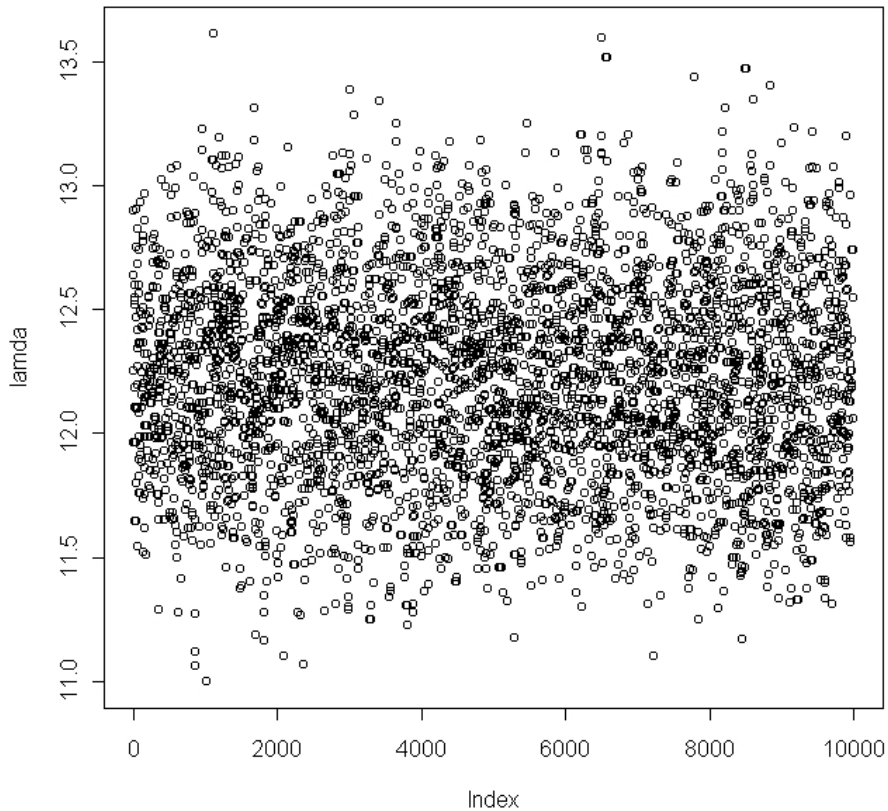
Now, Let's do a Larger MCMC Sampling

- **We got extremely *Lucky* on our Tuning for Example 1**
  - **I picked the Starting Point *arbitrarily* - 1**
  - **I picked the Step Size *arbitrarily* - 1**
- **Let's Expand and Use some of Our Preliminary Results**
  - **Pick a Starting Point of 12.25 - the *midpoint***
  - **Pick a Step Size a little larger to reduce acceptances - 1.5** ← ***Not Necessary, Just to Show the Effect***

```
> runexample1(nsamps=10000,lamda0=12.25,dlamda0=1.5)
Example 1:
  Number of Samples:   10000
  Initial Lamda:       12.25
  Initial Lamda Step:  1.5
  Acceptance Ratio:    0.3867387
> plot(lamda)
> phist(lamda)
> phist(lamda,mainlab="Large Sample Lamda Density",
+ xname="lamda" ,bwadj=1.5)
> pxgta(lamda,13)
[1] 0.0195
```

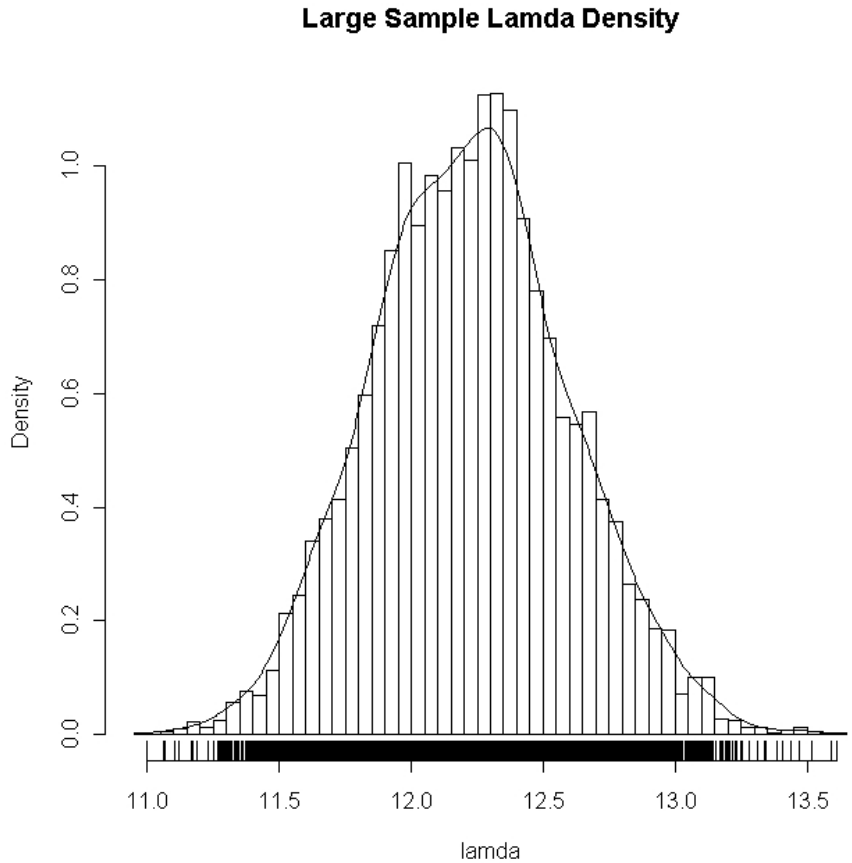
Our New MCMC Samples

- **Our *Acceptance Ratio* is Lower now due to Increased Step Size, but still in Range and okay - 39%**
- ***Intelligent Starting Point* Eliminated need for Burn-in**
- **Nice Visual Noise Characteristics**



Our New MCMC Posterior Uncertainty Model

- **Looks Much the Same**
  - **Smoother**
  - **Fewer Jags in Histogram**
  - **More Noticeable Skew to Lower Values**
- **Larger Samplings would Probably not Improve much on This**
- **Probability that Manager is Right about needing new staff is now 1.95% -**
  Still No New Staff!



Comparing Classical and MCMC Results

- **Classical Results and Our *PRA/MCMC* Results Produced the Same Decision**
- **There are some *Huge* Differences However**
  - **Classical *p*-value (99.7% - the frequency that if we were able to repeat the experiment an infinite number of times, that we would get this data set or worse given that the null hypothesis is true) was Much Larger than our Significance Level α = 0.1**

    What does this Tell us Quantitatively?
  - **Our *PRA/MCMC* Results gave us a Probability (only 1.95%) *(given the data that we actually have)* that the Manager was Right about needing New Staff**

    This is Quantitative!
- **Which would you want to report to Upper Management?**

A Few Notes on this Example

- **This also is one case where an *Ignorance Prior* is a *Conjugate Prior* and Produces a Named and Recognizable Posterior Uncertainty Model**
- **Our Posterior should be recognizable as a *Gamma* Model**

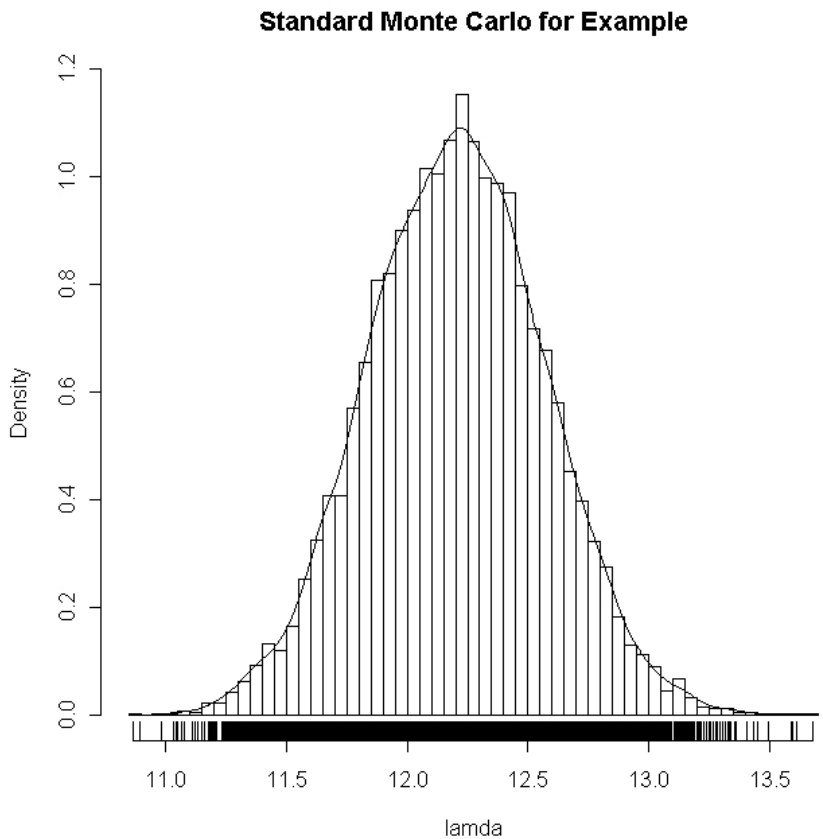
$$pd\left(\lambda \mid data, H\right) \propto \lambda^{-\frac{1}{2}} e^{-n\lambda} \prod_{i=1}^{n} \frac{\lambda^{datum_i}}{\Gamma\left(datum_i + 1\right)} \propto \lambda^{-\frac{1}{2}+\sum_{i=1}^{n} datum_i} e^{-n\lambda}$$

$$pd\left(\lambda \mid data, H\right) \sim Gamma\left(\lambda \mid \sum_{i=1}^{n} datum_i - \frac{1}{2}, n\right)$$

Let's Look at a Standard Monte Carlo Solution

- **Very Easy to do in R**
- **Don't Even need to do Standard Monte Carlo since we Have a Known Posterior**
- **Results are *all about the Same* - 1.5% → 1.95%**

```
> n<-length(data)
> alpha<-sum(data)-1/2
> lamda<-rgamma(10000,alpha,n)
> phist(lamda, xname="lamda“,
+mainlab="Standard Monte Carlo for Example")
> pxgta(lamda,13)
[1] 0.0153
> 1-pgamma(13,alpha,n)
[1] 0.01691248
```



Example #1 Synopsis

- **This Example Demonstrates**
  - **Classical vs. *PRA/MCMC* Decision Results**
  - ***Incredible Simplicity* of MCMC M-H Algorithm and Code**
  - **Ease of Tuning**
- **Codes as used in Example are Available on my Faculty Website**
  - **Data of Course (Used in HW06)**
  - **Utilities**
  - **Example Functions for Log Posterior and M-H Algorithm**
- **Next Module: Another Yet More Complex Example**

First, a Word about MCMC Coding Simplifications in R

- **Recall the Log Posterior for Example #1 and it's Code**

$$pd\left(\lambda \mid data, H\right) \propto \lambda^{-\frac{1}{2}} e^{-n\lambda} \prod_{i=1}^{n} \frac{\lambda^{datum_i}}{\Gamma\left(datum_i + 1\right)}$$

$$Log\left(pd\left(\lambda \mid data, H\right)\right) \propto -\frac{1}{2} Log\left(\lambda\right) - n\lambda + \sum_{i=1}^{n} \left[datum_i Log\left(\lambda\right) - Log\left(\Gamma\left(datum_i + 1\right)\right)\right]$$

```
lpd<-function(lamda,data) {  # Log of our Posterior
  n<-length(data)  # Get number of data
  sum<-0
  for (i in 1:n){  # Sum over data terms
    sum<-sum+data[i]*log(lamda)-lgamma(data[i]+1)
  }
  temp<-sum-log(lamda)/2-n*lamda  # Compute log posterior
  return(temp)
}
```

- **Recall that Log(*Posterior*) is the Log of the Product of *Likelihood* and *Prior*, it becomes Log(*Likelihood*) + Log(*Prior*)**

Our Posterior R Code could have been Much Simpler

- **Some *Simpler* R Coding Alternatives**

```
lpd<-function(lamda,data) {  # Log of our Posterior
  return(sum(data*log(lamda)-lgamma(data+1))-log(lamda)/2-length(data)*lamda)
}
```

```
lpd<-function(lamda,data) {  # Log of our Posterior
  return(sum(dpois(data,lamda,log=TRUE))-log(lamda)/2) # Compute log posterior
}
```

- **A Slightly More Complex, but More *General* Alternative**

```
llike<-function(data,lamda) {  # Log of our Likelihood
  return(sum(dpois(data,lamda,log=TRUE))) # Compute log likelihood
}
lprior<-function(lamda) { # Log of our Prior
  return(-log(lamda)/2) # Compute log prior
}
lpd<-function(lamda,data) {  # Log of our Posterior
  return(llike(data,lamda)+lprior(lamda)) # Compute log posterior
}
```

Example #2

- **From Homework 06, Hayter Problem 6.5.4, Figure 6.60 for data**
  - **Data: Heights of Adult Males to visit a Clinic over One Week**
  - **A *Reasonable* Uncertainty Model for this Data could be the *Lognormal* Model**
- **Hayter Problem 8.4.5 - For this Data, Using 90, 95, and 99% Confidence Intervals, Could an Average Height of 70 inches be *Plausible*?**

Classical Solution for Example #2

- **We Need to Compute the 90, 95, and 99% *Confidence Intervals***
- **If these Intervals contain 70 inches (our Hypothesis), We cannot reject the Hypothesis that the True Mean Height is 70 inches**
- **We Can Use Hayter's *Recipe* on Pg. 359 for Two-Sided t-Intervals, *Very Easy using R***
- **70 Inches is inside the 95 and 99% CI's, but not the 90% - *What are we to Decide?***

```
> # "Adult Male Heights in inches"
> data<-c(71.4,70.2,68.1,70.0,67.2,67.5,70.1,
+ 68.1,70.0,70.3,69.1,71.5,69.3,68.8,73.5,69.3,
+ 69.9,69.7,69.9,68.6,70.5,69.5,71.1,71.1,68.1,
+ 68.3,69.4,70.9,70.0,69.0,70.5,69.3,69.0,69.4,
+ 68.3,70.0,68.6,68.7,75.9,68.8,69.4,68.9,68.6,
+ 66.9,69.7,68.3,68.4,73.0,70.1,69.9,70.7,69.5,
+ 68.6,68.1,70.4,71.0,69.5,70.8,66.9,69.5)
> n<-length(data)
> xbar<-mean(data)
> s<-sqrt(var(data))
> alpha90<-c(0.05,0.95)
> alpha95<-c(0.025,0.975)
> alpha99<-c(0.005,0.995)
> ci90<-xbar+qt(alpha90,n-1)*s/sqrt(n)
> ci95<-xbar+qt(alpha95,n-1)*s/sqrt(n)
> ci99<-xbar+qt(alpha99,n-1)*s/sqrt(n)
> cat("90% CI:  (",ci90[1],",",ci90[2],")","\n")
90% CI:  ( 69.28974 , 69.94692 )
> cat("95% CI:  (",ci95[1],",",ci95[2],")","\n")
95% CI:  ( 69.48489 , 70.01179 )
> cat("99% CI:  (",ci99[1],",",ci99[2],")","\n")
99% CI:  ( 69.09495 , 70.14172 )
```

The PRA Solution

- **This Example at First Seems Much More Complex than Example #1**
  - **We Have *Two Parameters* for the Lognormal Model, μ and σ^2, the Parameters for the Underlying Normal Model**
  - **We are Concerned about the *Uncertainty Model for the Average* Height of Males, which is computed from the Lognormal Model Parameters**

$$mean = e^{\left(\mu + \frac{\sigma^2}{2}\right)}$$

  - **We don't Yet know How to Get this Uncertainty Model!**
- **First, We have to Choose a Prior Uncertainty**
  - **Let's Use Independent Ignorance (Jeffrey's) Priors**
  - $pd\left(\mu, \sigma^2 \mid H\right) \propto pd\left(\mu \mid H\right) pd\left(\sigma^2 \mid H\right) \propto 1 \times \frac{1}{\sigma^2} = \frac{1}{\sigma^2}$

Our Likelihood and Posterior

- **Our Likelihood is Simple, but Improper**

$$pd\left(data \mid \mu, \sigma^2, H\right) = \prod_{i=1}^{n}\left[\left(\sqrt{2\pi}\sigma * datum_i\right)^{-1} e^{-\frac{\left(Log(datum_i - \mu\right)^2}{2\sigma^2}}\right]$$

$$\propto \prod_{i=1}^{n}\left[\left(datum_i\right)^{-1} e^{-\frac{\left(Log(datum_i - \mu\right)^2}{2\sigma^2}}\right]$$

- **Our Posterior is Also Simple, but Improper**

$$pd\left(\mu, \sigma^2 \mid data, H\right) \propto \frac{1}{\sigma^2}\prod_{i=1}^{n}\left[\left(datum_i\right)^{-1} e^{-\frac{\left(Log(datum_i - \mu\right)^2}{2\sigma^2}}\right]$$

Coding our Posterior for MCMC

- **Now, Let's code this the More *General* Way in R**

```
llike<-function(dat,muc,varc) { # Log of Likelihood
  return(sum(dlnorm(dat,muc,sqrt(varc),log=TRUE)))
}
lprior<-function(muc,varc) { # Log of Prior
  return(-log(varc))
}
lpd<-function(muc,varc,dat) {  # Log of our Posterior
  return(llike(dat,muc,varc)+lprior(muc,varc))
}
```

Coding the M-H Algorithm - Coding Samplers

- Let's First Build Sampler Functions for μ and σ^2
- We will Use an *Internal Loop* over both Samplers in our M-H Code
- Here is the M-H Sampler function for μ

```
samplemu<-function(dat,muc,varc,dmuc) { # mu M-H Sampler
  pmu<-muc+runif(1,-dmuc,dmuc)  #  Get Proposed Sample
  logalpha<-lpd(pmu,varc,dat)-lpd(muc,varc,dat)
  logu<-log(runif(1,0,1))  #  Get log uniform sample for acceptance
  if (logu<logalpha) {  # Accepted proposed sample
    mu<<-append(mu,pmu)  #  Add proposed sample to vector
    mucurr<<-pmu  #  Reset current to proposed sample
    muacc<<-muacc+1  #  Count acceptance
  }
  else { #  Did not accept proposed Sample
    mu<<-append(mu,muc)  # Add current sample to vector
  }
}
```

Coding the σ^2 Sampler

- Here is the Sampler Function for σ^2
- A bit More Complex than the μ Sampler Function due to the Possibility of Illegal Proposals

```
samplevar<-function(dat,muc,varc,dvarc) { # var M-H Sampler
  pvar<-varc+runif(1,-dvarc,dvarc)  #  Get Proposed Sample
  if (pvar>0) { # Got a legal proposal (var>0 is required)
    logalpha<-lpd(muc,pvar,dat)-lpd(muc,varc,dat)
    logu<-log(runif(1,0,1))  #  Get log uniform sample, acceptance
    if (logu<logalpha) {  # Accepted proposed sample
      var<<-append(var,pvar)  #  Add proposed sample to vector
      varcurr<<-pvar  #  Reset current to proposed sample
      varacc<<-varacc+1  #  Count acceptance
    }
    else { #  Did not accept proposed Sample
      var<<-append(var,varc)  # Add current sample to vector
    }
  }
  else { # Got an illegal proposal, do not accept
    var<<-append(var,varc)  # Add current sample to vector
  }
}
```

Coding the M-H Loop

- Now the Actual M-H Loop becomes a little Simpler
- Functions are the way to go in R
- Note the Default Initial μ and σ^2 values and Steps

```
runexample2<-function(nsamps=3000,dat=data,
  mu0=mean(log(data)),dmu0=mu0/2,
  var0=var(log(data)),dvar0=var0/2) {
  #  M-H for Example 2, Initialize the Algorithm
  mu<<-mu0  #  Set first mu sample to Starting Value
  mucurr<<-mu0  #  Set current mu sample to Starting Value
  dmu<-dmu0
  var<<-var0  #  Set first var sample to Starting Value
  varcurr<<-var0  #  Set current var sample to Starting Value
  dvar<-dvar0
  muacc<<-0  #  Zero mu Acceptance Counter
  varacc<<-0  #  Zero var Acceptance Counter
  for (i in 2:nsamps) {  #  The M-H loop
    samplemu(dat,mucurr,varcurr,dmu)
    samplevar(dat,mucurr,varcurr,dvar)
  }
  cat("Example 2:","\n") #  Print out some Useful Data
  cat("  Number of Samples:    ",nsamps,"\n")
  cat("  Initial mu and step:  ",mu0,dmu0,"\n")
  cat("  Initial var and step: ",var0,dvar0,"\n")
  cat("  mu Acceptance Ratio:  ",muacc/(nsamps-1),"\n")
  cat("  var Acceptance Ratio: ",varacc/(nsamps-1),"\n")
}
```

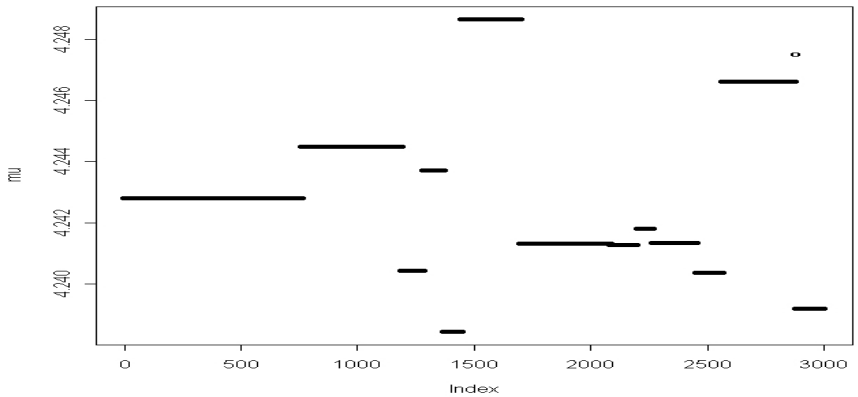
Running the M-H Algorithm

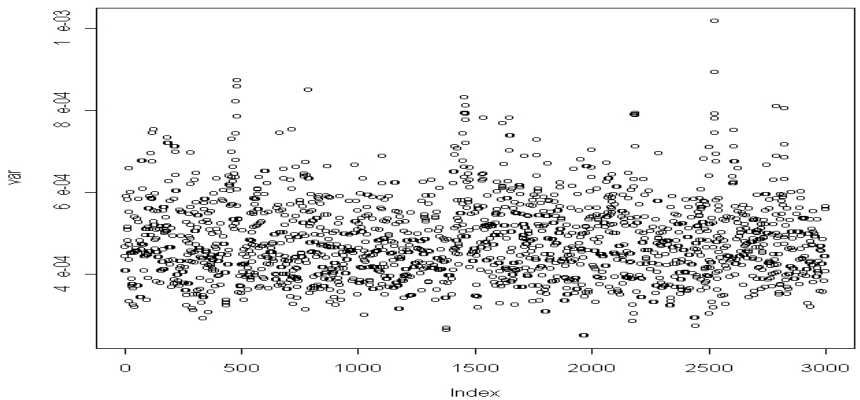
- **Copy into the R console**
  - **MCMC Utilities from my Faculty Website**
  - **Our Log Posterior, Log Likelihood, and Log Prior Functions**
  - **Our M-H Algorithm and Sampler Functions**
  - **Our Data (figure 6.60 data)**
- **Now Run the MCMC Sampler**

```
> runexample2()
Example 2:
  Number of Samples:     3000
  Initial mu and step:   4.242797 2.121398
  Initial var and step:  0.000465831 0.0002329155
  mu Acceptance Ratio:   0.004334778
  var Acceptance Ratio:  0.4988329
> plot(mu)
> plot(var)
```

Our Initial MCMC Results

- **First, We did not get Lucky with this Example**
  - **Our *Acceptance Ratio* for μ is way too low (0.004), outside our Desired Range of 30-60%**
  - **We need to *decrease* the step size a lot**
- **Here are the Sample plots for μ and σ^2**
- **Obviously, Way too much Tracking for μ**

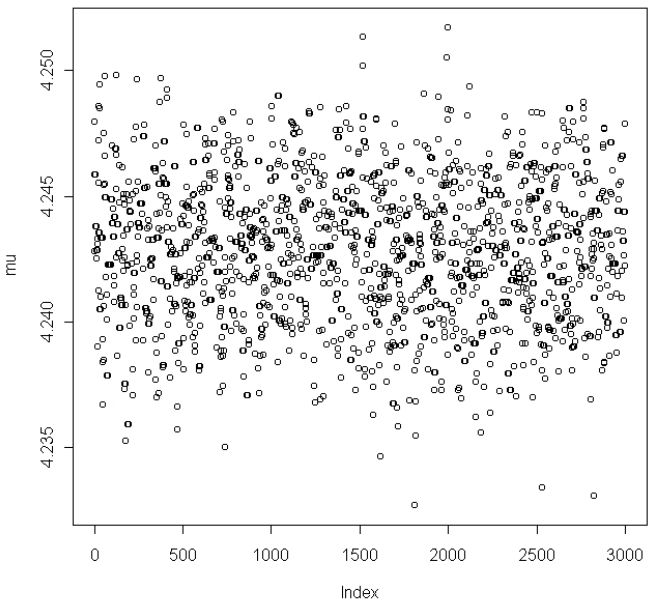


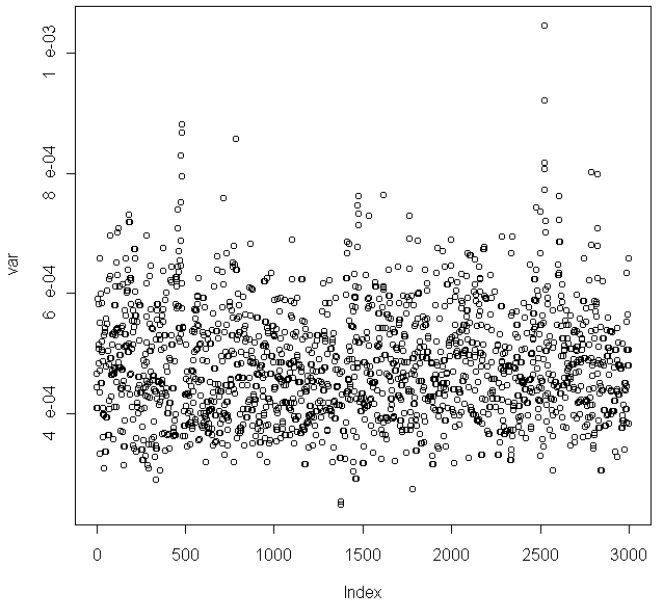


With Just a Little Tuning

- We Decrease the μ step size by more than 2 orders of Magnitude to get a Good μ Acceptance Ratio (empirically, many runs)
- Note that this changed the σ^2 Acceptance Ratio a little

```
> runexample2(dmu0=0.01)
Example 2:
  Number of Samples:     3000
  Initial mu and step:   4.242797 0.01
  Initial var and step:  0.000465831 0.0002329155
  mu Acceptance Ratio:   0.4248083
  var Acceptance Ratio:  0.5068356
> plot(mu)
> plot(var)
```

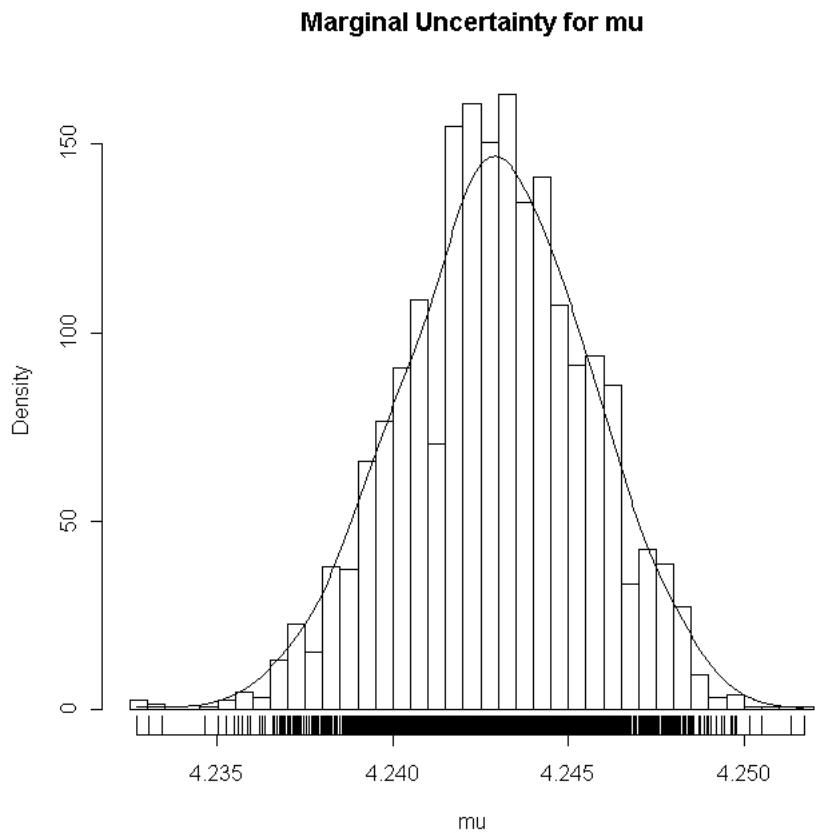




Let's Look at our μ Marginal

- **We Simply use the phist function from the Utilities on the μ samples to look at the μ Marginal Uncertainty Model**

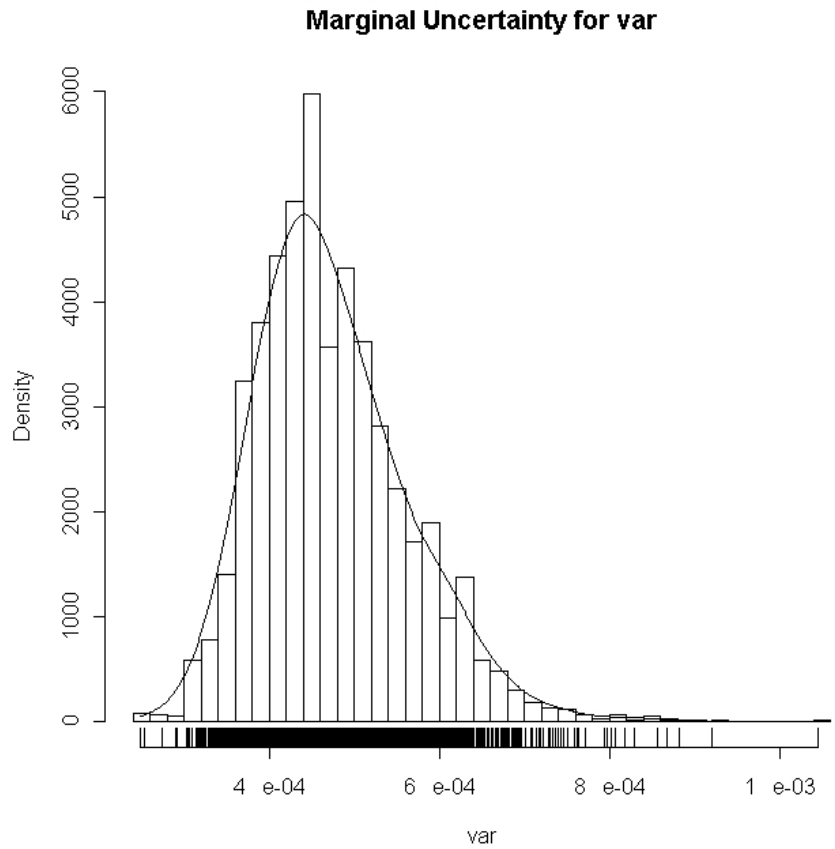
```
> phist(mu,xname="mu",bwadj=1.8,
+ mainlab="Marginal Uncertainty for mu")
```



Let's Look at our σ^2 Marginal

- **We Simply use the *phist* function from the Utilities on the σ^2 samples to look at the σ^2 Marginal Uncertainty Model**

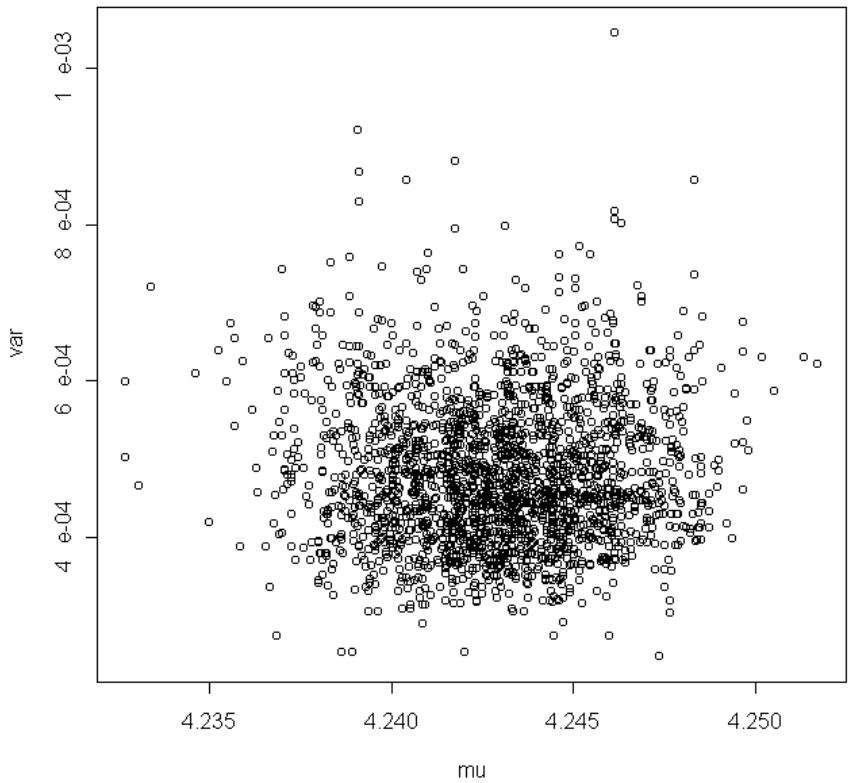
```
> phist(var,xname="var",bwadj=1.8,
+ mainlab="Marginal Uncertainty for var")
```



Let's Look at the Joint μ and σ^2 Uncertainty Model

- Each μ and σ^2 with a common index is a Joint Sample
- We Just Plot them Against Each Other
- The Density of the Points is the Density of the Joint Uncertainty of μ and σ^2

```
> plot(mu,var)
```



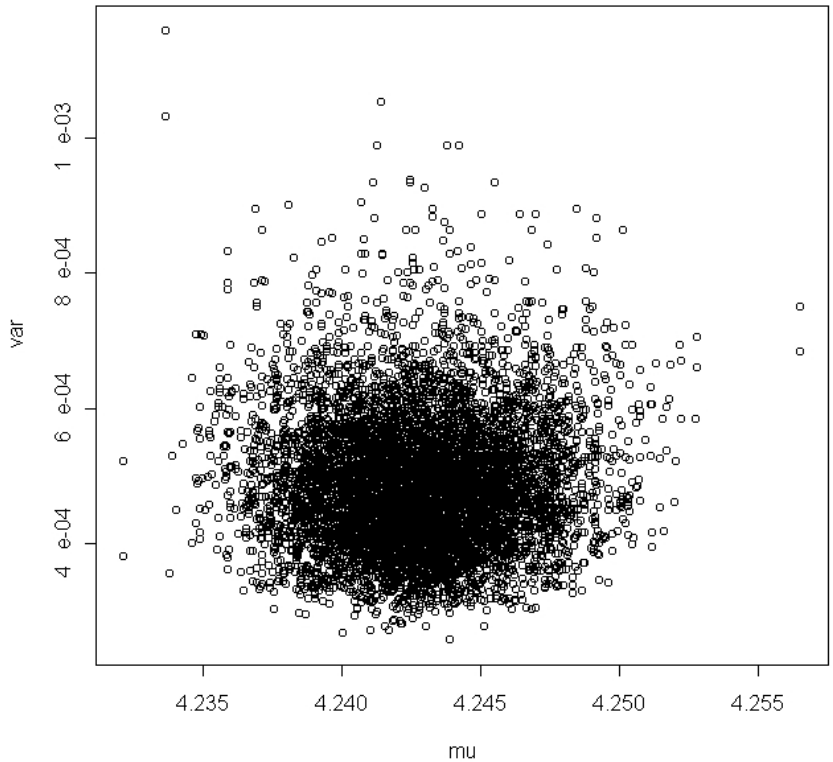
Now, Let's do a Larger MCMC Sampling

- **Our Tuning for Example 2 was Relatively Easy**
  - **I picked the Starting Points to be in the Middle**
  - **I picked the Initial Step Sizes rather arbitrarily, but only had to Change the μ step size to tune**
- **Our Acceptance Ratios Stay in Range Nicely Now for 10,000 Samples**

```
> runexample2(10000,dmu0=0.01)
Example 2:
  Number of Samples:     10000
  Initial mu and step:   4.242797 0.01
  Initial var and step:  0.000465831 0.0002329155
  mu Acceptance Ratio:   0.4308431
  var Acceptance Ratio:  0.5253525
> plot(mu,var)
```

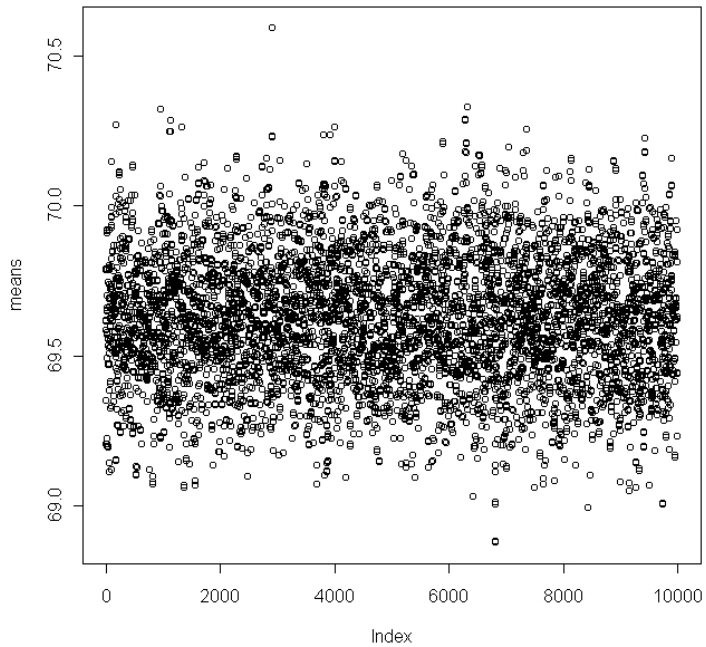
Our New Joint MCMC Samples

- **With 10,000 Joint Samples, the Joint Density is Obvious**
- **Note the few Isolated Joint Samples that would Normally be considered *Outliers***
- **Even More Samples (e.g. 10 million) would show even More "*Outliers*"**



Recall our Problem

- **Our Problem had to do with the *Plausibility* of an Average Height of 70 inches**
- **Each of our Joint Samples of μ and σ^2 defines a Possible Uncertainty Model for Height, with a Calculable Average**
- **We Need the Uncertainty Model for Average Height based on our Data**
- **Getting Samples of the Uncertainty Model for Average Height is Actually Very Easy with MCMC**
- **We Simply Evaluate the Formula for the Lognormal Model Average at Each Joint μ and σ^2 Sample**

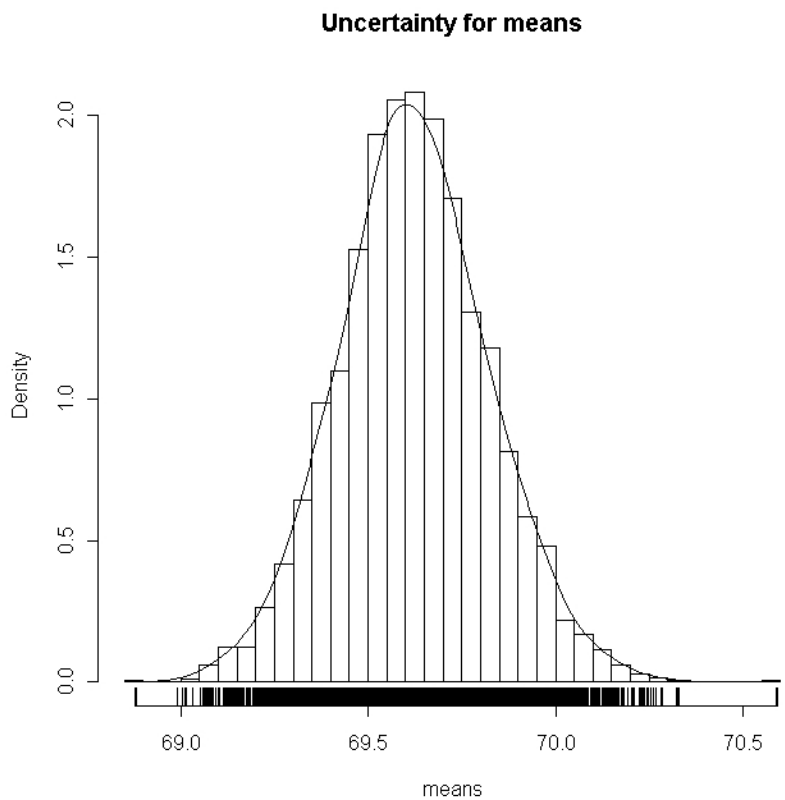



```
> means<-exp(mu+var/2)
> plot(means)
```

Our Posterior Uncertainty Model for Average Height

- We now have very good Sampling of the *Average Height Uncertainty Model*
- We can use the *phist* function to look at the Posterior Average Height Uncertainty Model
- Using the *pxlta* function from the Utilities, we find that the Probability that the Average Height given our Data is less than 70 inches is about 97%

```
> phist(means,xname="means",bwadj=1.8,
+ mainlab="Uncertainty for means")
> pxlta(means,70)
[1] 0.9693
```



Comparing Classical and PRA/MCMC Results

- **With our Classical Results using Confidence Intervals, it really was not clear whether an Average Height of 70 inches was *Plausible* or not**
  - **70 inches was inside the 95% and 99% Intervals**
  - **70 inches was not inside the 90% Interval**
  - **Recall what our Confidence Intervals are: Given that the Hypothesis is really true (Average Height is 70 inches), X% (90, 95, or 99%) of the time we compute a new Interval from a new Batch of data, the Interval will contain 70 inches**
  - **Is this Batch of Data one of the 10% that will produce a 90% Interval that will not contain 70 inches, or should we Reject the Hypothesis that the Average Height is really 70 inches?**
- **With our *PRA/MCMC* Results, we found that there was a 97% probability based on our 60 Data that the Average Height is less than 70 inches**
- ***Which would you want to report to Upper Management?***

A Few Notes on this Example

- **This example Demonstrated a few Significant and Easy Capabilities of MCMC**
  - **We had good sampling of a complex Joint Uncertainty Model**
  - **We were able to Develop an Uncertainty Model for a function of our Joint Posterior Model rather Easily**
  - **Tuning was rather Easy**
- **This Problem was *not solvable* without MCMC**

Example #2 Synopsis

- **This Example Demonstrates**
  - **Joint Multivariate Uncertainty Models**
  - **Easy Development of Uncertainty Models for functions of the Joint Multivariate Uncertainty Models**
  - **More Ease of Tuning**
- **Codes as used in Example are Available on my Faculty Website**
  - **Data of Course (Used in HW06)**
  - **Utilities**
  - **Example Functions**
- **Next Module: Advanced MCMC Capabilities**
- **Homework 09 is Due before Module 27 is Released**