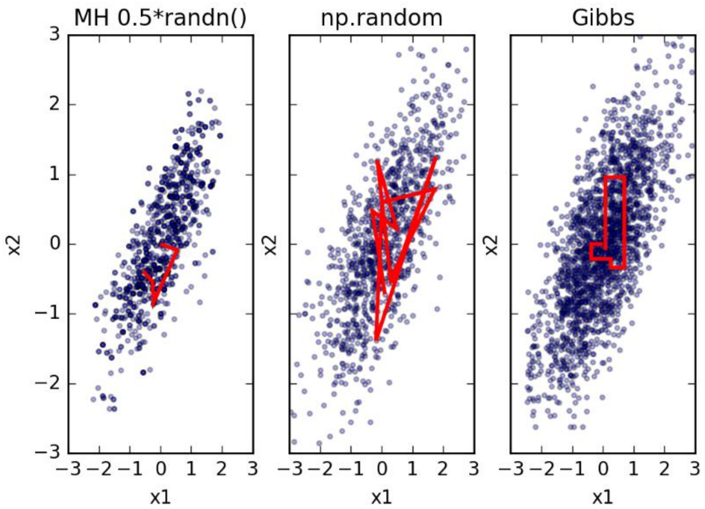
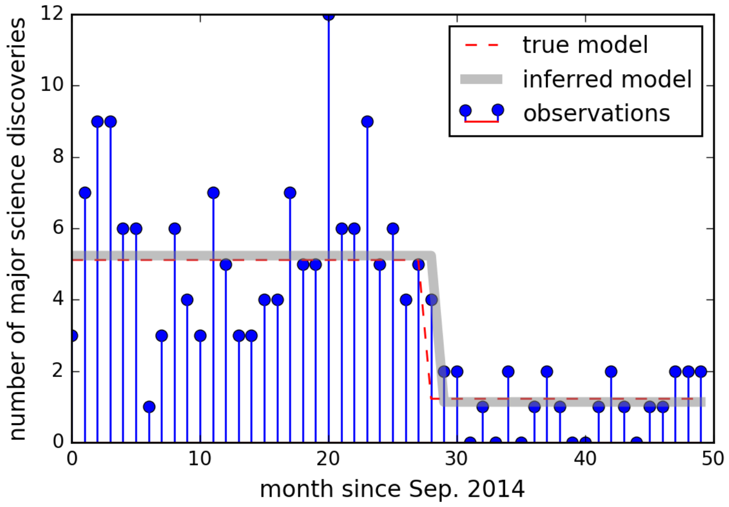
The Gibbs Sampling Algorithm: with Applications to Change-point Detection and Restricted Boltzmann Machine

Yubo “Paul” Yang

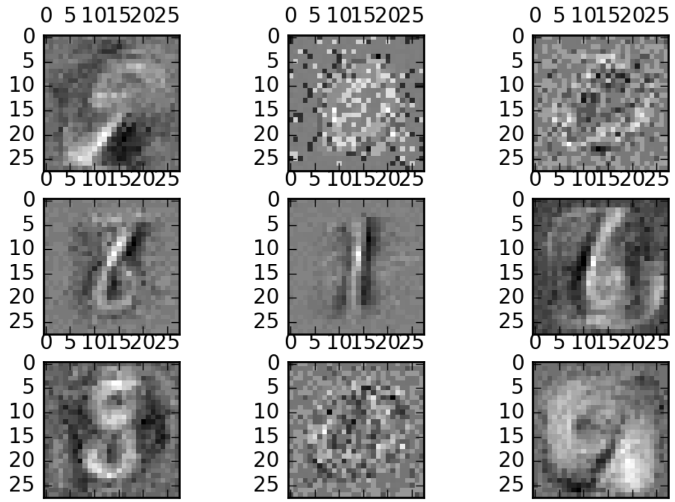
Jan. 24 2017

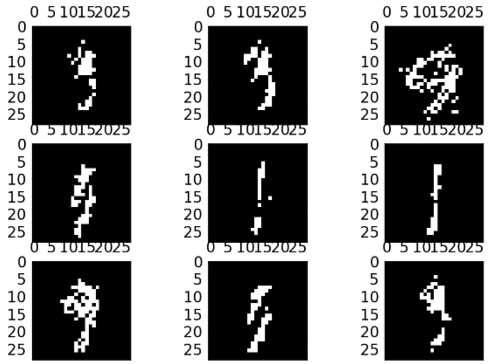


Change-point model



Restricted Boltzmann machine





Introduction: History

Donald Geman
@ Johns Hopkins

- 1965 B.A. in English Literature from UIUC
- 1970 Ph.D in Mathematics from Northwestern

Stuart Geman
@ Brown

- 1971 B.S. in Physics from UMich
- 1973 MS in Neurophysiology from Dartmouth
- 1977 Ph.D in Applied Mathematics from MIT

- 1984 Gibbs Sampling (IEEE Trans. Pattern Anal. Mach. Intell, 6, 721-741, 1984.)
- 1986 Markov Random Field Image Models (PICM. Ed. A.M. Gleason, AMS, Providence)
- 1997 Decision Trees and Random Forest (Neural Computation., 9, 1545-1588, 1997) with Y. Amit

Gibbs Sampling: One variable at a time

Basic version:

- One variable at a time
- Special case of Metropolis-Hasting (MH)

i.e. Acceptance = 1

*Block version***:**

- Sample all independent variables simultaneously

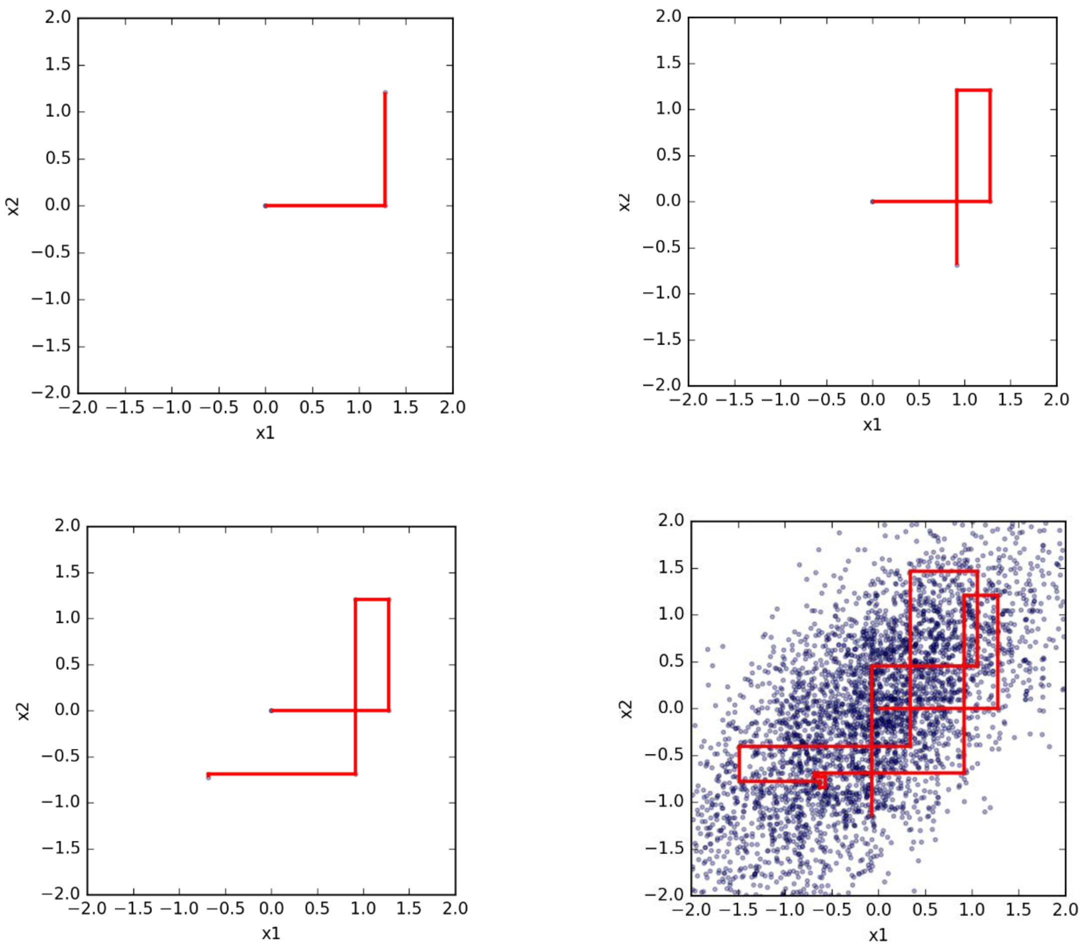
Collapsed version:

- Trace over some variables (i.e. ignore them)

Samplers within Gibbs:

- Eg. Sample some variables with MH

Basic Gibbs sampling from bivariate Normal



Basic Example: Sample from Bivariate Normal Distribution

Example inspired by: MCMC: The Gibbs Sampler
, *The Clever Machine*,
https://theclevermachine.wordpress.com/2012/11/05/mcmc-the-gibbs-sampler/

Q0/ How to sample x from standard normal distribution $\mathrm{N}(\mu = 0, \sigma = 1)$?

Basic Example: Sample from Bivariate Normal Distribution

Example inspired by: MCMC: The Gibbs Sampler
, *The Clever Machine*,
https://theclevermachine.wordpress.com/2012/11/05/mcmc-the-gibbs-sampler/

Q0/ How to sample x from standard normal distribution $\mathcal{N}(\mu = 0, \sigma = 1)$?

A0/ np.random.randn() samples from P(x) = $\frac{1}{\sqrt{2\pi\sigma^2}}$ exp$[-\frac{(x-\mu)^2}{2\sigma^2}]$.

Bivariate normal distribution is the generalization of the normal distribution to two variables:

$$P(x_1, x_2) = \mathcal{N}(\mu_1, \mu_2, \Sigma) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} exp\left[-\frac{z}{2(1-\rho^2)}\right]$$

Basic Example: Sample from Bivariate Normal Distribution

Example inspired by: MCMC: The Gibbs Sampler
, *The Clever Machine*,
https://theclevermachine.wordpress.com/2012/11/05/mcmc-the-gibbs-sampler/

Q0/ How to sample x from standard normal distribution $\mathcal{N}(\mu = 0, \sigma = 1)$?

A0/ np.random.randn() samples from P(x) = $\frac{1}{\sqrt{2\pi\sigma^2}} \exp[-\frac{(x-\mu)^2}{2\sigma^2}]$.

Bivariate normal distribution is the generalization of the normal distribution to two variables:

$$P(x_1, x_2) = \mathcal{N}(\mu_1, \mu_2, \Sigma) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} exp\left[-\frac{z}{2(1-\rho^2)}\right]$$

where $\Sigma = \begin{pmatrix} \sigma_1 & \rho \\ \rho & \sigma_2 \end{pmatrix}$ and $z = \frac{(x_1-\mu_1)^2}{\sigma_1^2} - \frac{2\rho(x_1-\mu_1)(x_2-\mu_2)}{\sigma_1\sigma_2} + \frac{(x_2-\mu_2)^2}{\sigma_2^2}$

For simplicity, let $\mu_1 = \mu_2 = 0$, and $\sigma_1 = \sigma_2 = 1$ then:

$$\ln\ P(x_1, x_2) = -\frac{x_1^2 - 2\rho x_1 x_2 + x_2^2}{2(1-\rho^2)} + const.$$

Q/ How to sample $\boldsymbol{x_1}, \boldsymbol{x_2}$ from $\boldsymbol{P(x_1, x_2)}$?

Basic Example: Sample from Bivariate Normal Distribution

The joint probability distribution of x_1, x_2 has log:

$$\ln\ P(x_1, x_2) = -\frac{x_1^2 - 2\rho x_1 x_2 + x_2^2}{2(1-\rho^2)} + const.$$

Q/ How to sample $\boldsymbol{x_1}, \boldsymbol{x_2}$ from $\boldsymbol{P}(\boldsymbol{x_1}, \boldsymbol{x_2})$?
A/ Gibbs sampling.
Fix x2, sample x1 from $\boldsymbol{P}(\boldsymbol{x_1}|\boldsymbol{x_2})$
Fix x1, sample x2 from $\boldsymbol{P}(\boldsymbol{x_2}|\boldsymbol{x_1})$
Rinse and repeat

Basic Example: Sample from Bivariate Normal Distribution

The joint probability distribution of x_1, x_2 has log:

$$\ln\ P(x_1, x_2) = -\frac{x_1^2 - 2\rho x_1 x_2 + x_2^2}{2(1-\rho^2)} + const.$$

Q/ How to sample $\boldsymbol{x_1, x_2}$ from $\boldsymbol{P(x_1, x_2)}$?
A/ Gibbs sampling.
Fix x2, sample x1 from $\boldsymbol{P(x_1|x_2)}$
Fix x1, sample x2 from $\boldsymbol{P(x_2|x_1)}$
Rinse and repeat

The full conditional probability distribution of x_1 has log:

$$\ln P(x_1|x_2) = -\frac{x_1^2 - 2\rho x_1 x_2}{2(1-\rho^2)} + const. = -\frac{(x_1 - \rho x_2)^2}{2(1-\rho^2)} + const. \Rightarrow$$

$$P(x_1|x_2) = \mathrm{N}(\mu = \rho x_2, \sigma = \sqrt{1-\rho^2})$$

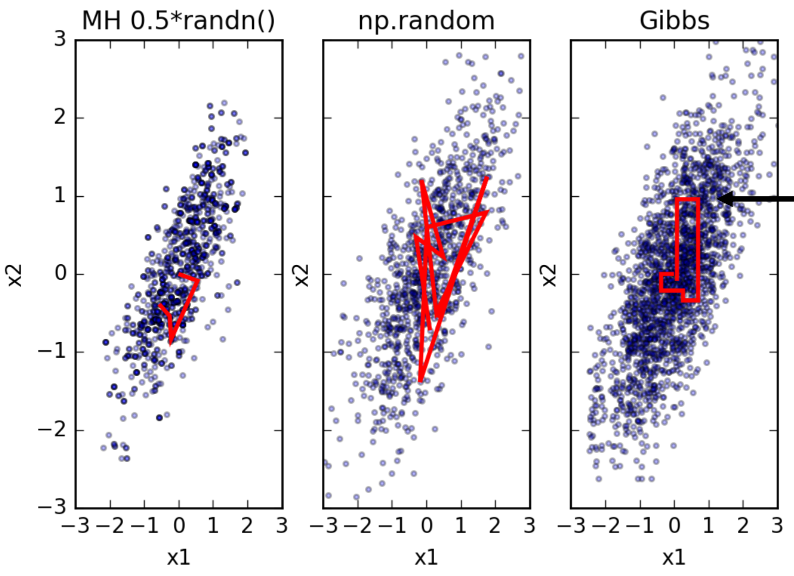
```
new_x1 = np.sqrt(1-rho*rho) * np.random.randn() + rho*x2
```

Basic Example: Sample from Bivariate Normal Distribution

```python
def gibbs_bivariate_std_normal(rho,nsample):
    sample0 = (0,0)
    samples = np.zeros([2*nsample,2])
    samples[0,:] = sample0
    for isample in range(1,nsample):
        # start with previous sample
        samples[2*isample,:] = samples[2*isample-1,:]
        # resample first element
        samples[2*isample,0] = np.sqrt(1-rho*rho)*np.random.randn() + rho*samples[2*isample-1,1]

        # start with previous sample
        samples[2*isample+1,:] = samples[2*isample,:]
        # resample second element
        samples[2*isample+1,1] = np.sqrt(1-rho*rho)*np.random.randn() + rho*samples[2*isample,0]
    # end for isample
    return samples
# end def
```

$\rho = 0.8$

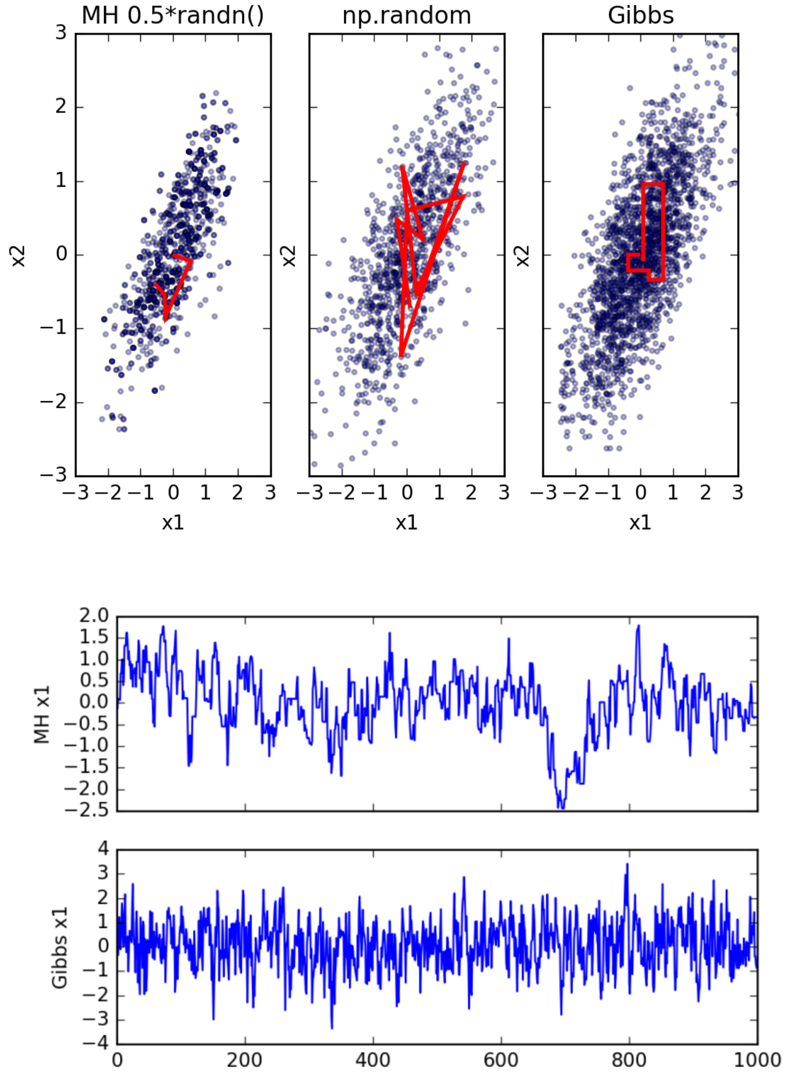


Fixing x2 shifts the mean of x1 and changes its variance

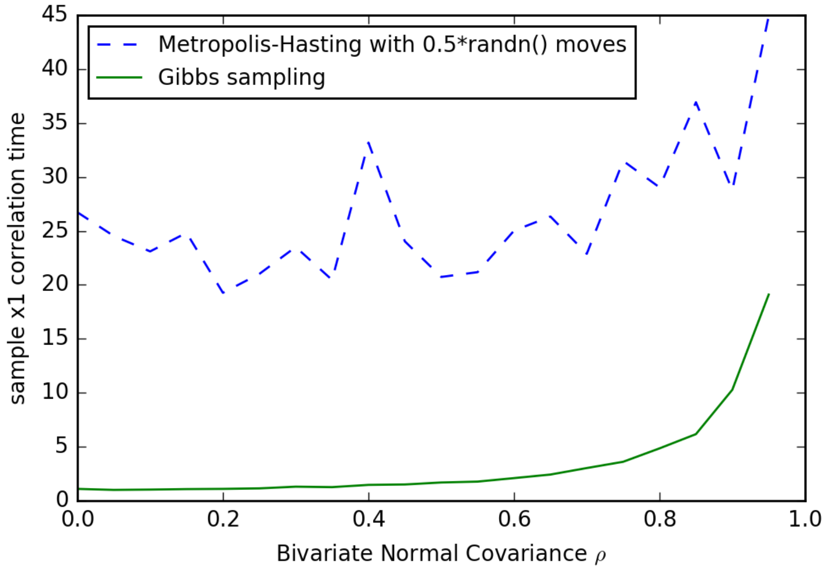
Basic Example: Sample from Bivariate Normal Distribution

Gibbs sampler has worse correlation than numpy's built-in multivariate_normal sampler,

but is much better than naïve Metropolis (reversible moves, $A = \min(1, \frac{P(\boldsymbol{x'})}{P(\boldsymbol{x})})$)



Both Gibbs and Metropolis still fail when correlation is too high.

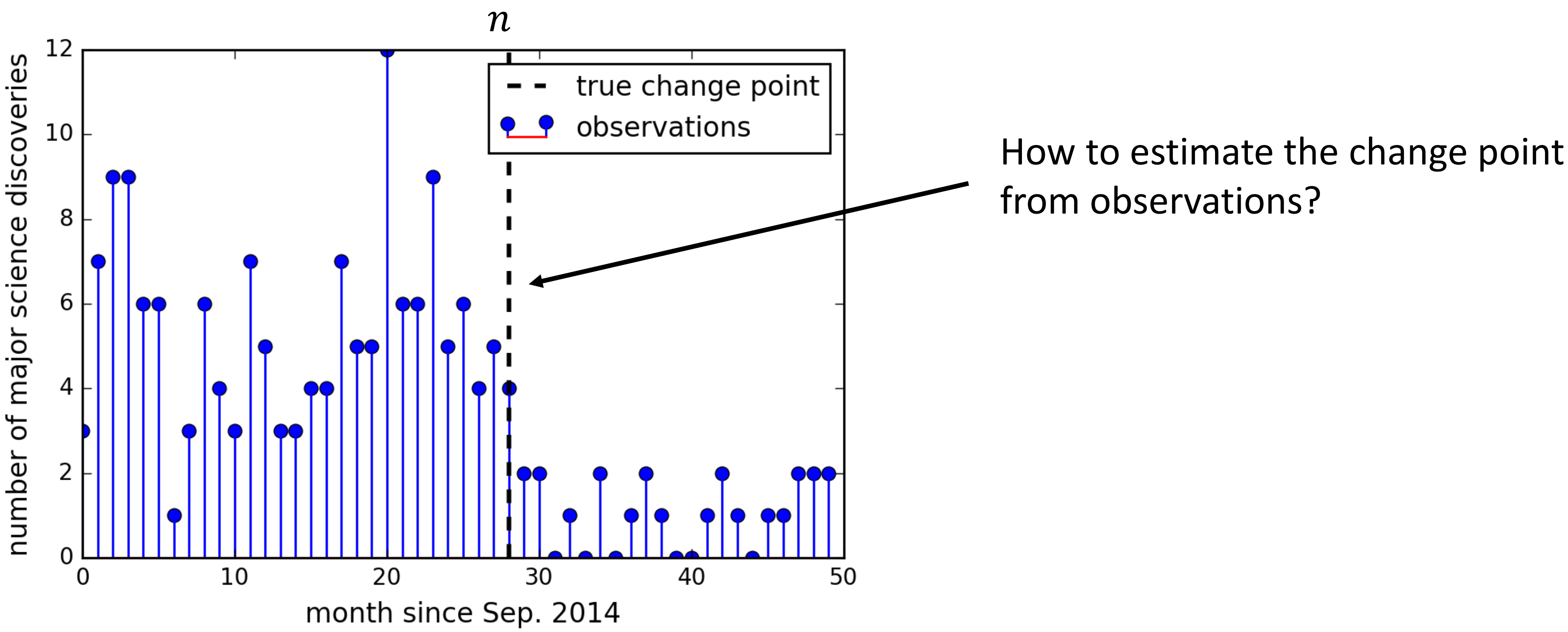


Model Example: Train a Change-point Model with Bayesian Inference

Example inspired by: Ilker Yildirim's notes on Gibbs sampling, http://www.mit.edu/~ilkery/papers/GibbsSampling.pdf

Bayesian Inference: Improve 'guess' model with data.

The question that change-point model answers:
when did a change occur to the distribution of a random variable?



Model Example: Train a Change-point Model with Bayesian Inference

Example inspired by: Ilker Yildirim's notes on Gibbs sampling, http://www.mit.edu/~ilkery/papers/GibbsSampling.pdf

- change-point model: a particular probability distribution of observables and model parameters

(Gamma prior, Poisson posterior)

$$P(x_0, x_1, \dots, x_{N-1}, \lambda_1, \lambda_2, n) = \prod_{i=0}^{n-1} Poisson(x_i, \lambda_1) \prod_{i=n}^{N-1} Poisson(x_i, \lambda_2)$$

$$Gamma(\lambda_1; a=2, b=1) Gamma(\lambda_2; a=2, b=1) Uniform(n, N)$$

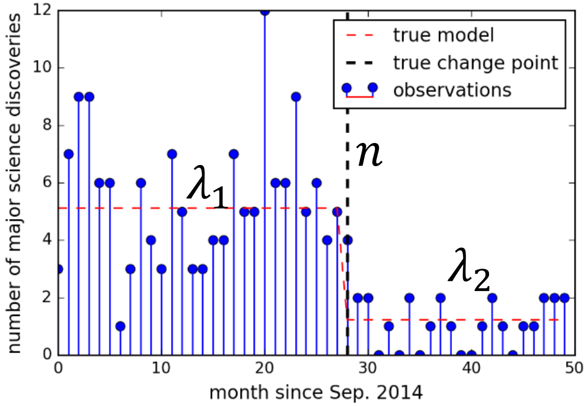
where

$$Poisson(x; \lambda) = e^{-\lambda} \frac{\lambda^x}{x!}$$

$$Gamma(\lambda; a, b) = \frac{1}{\Gamma(a)} b^a \lambda^{a-1} \exp(-b\lambda)$$

$$Uniform(n; N) = 1/N$$

Q/ What is the full conditional probability of λ_1?



Model Example: Train a Change-point Model with Bayesian Inference

Example inspired by: Ilker Yildirim's notes on Gibbs sampling, http://www.mit.edu/~ilkery/papers/GibbsSampling.pdf

- change-point model: a particular probability distribution of observables and model parameters

(Gamma prior, Poisson posterior)

$$P(x_0, x_1, \dots, x_{N-1}, \lambda_1, \lambda_2, n) = \prod_{i=0}^{n-1} Poisson(x_i, \lambda_1) \prod_{i=n}^{N-1} Poisson(x_i, \lambda_2)$$

$$Gamma(\lambda_1; a = 2, b = 1) Gamma(\lambda_2; a = 2, b = 1) Uniform(n, N)$$

where

$$Poisson(x; \lambda) = e^{-\lambda} \frac{\lambda^x}{x!}$$

$$Gamma(\lambda; a, b) = e^{-b\lambda} \frac{\lambda^{a-1}}{\Gamma(a)} \times b^a$$

$$Uniform(n; N) = 1/N$$

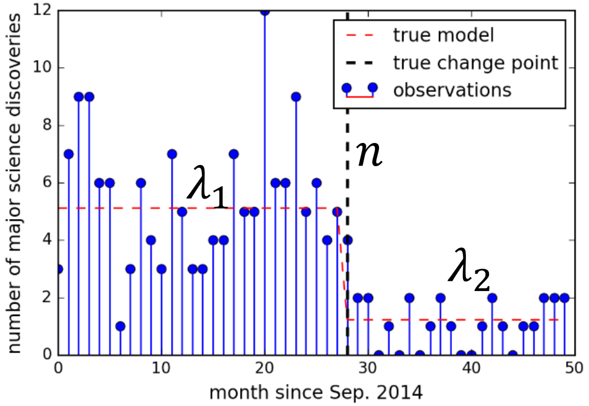
- Without observation, model parameters come from the *prior distribution* (the guess):

$$P(\lambda_1, \lambda_2, n) = Gamma(\lambda_1; a = 2, b = 1) Gamma(\lambda_2; a = 2, b = 1) Uniform(n, N)$$

- After observations, model parameters should be sampled from the *posterior distribution:*

$$P(\lambda_1, \lambda_2, n | x_0, x_1, \dots, x_{N-1})$$

Q/ How to sample from the joint posterior distribution of λ_1, λ_2, n?



Model Example: Train a Change-point Model with Bayesian Inference

Example inspired by: Ilker Yildirim's notes on Gibbs sampling, http://www.mit.edu/~ilkery/papers/GibbsSampling.pdf

Gibbs sampling require full conditionals

$$\ln P(\lambda_1|\lambda_2, n, \boldsymbol{x}) = \ln Gamma(\lambda_1; a + \sum_{i=0}^{n-1} x_i, b + n)$$

$$\ln P(\lambda_2|\lambda_1, n, \boldsymbol{x}) = \ln Gamma(\lambda_2; a + \sum_{i=n}^{N-1} x_i, b + N - n)$$

$$\ln P(n|\lambda_1, \lambda_2, \boldsymbol{x}) = \underline{mess(n|\lambda_1, \lambda_2, \boldsymbol{x})}$$

Q/How to sample this mess?!

Model Example: Train a Change-point Model with Bayesian Inference

Example inspired by: Ilker Yildirim's notes on Gibbs sampling, http://www.mit.edu/~ilkery/papers/GibbsSampling.pdf

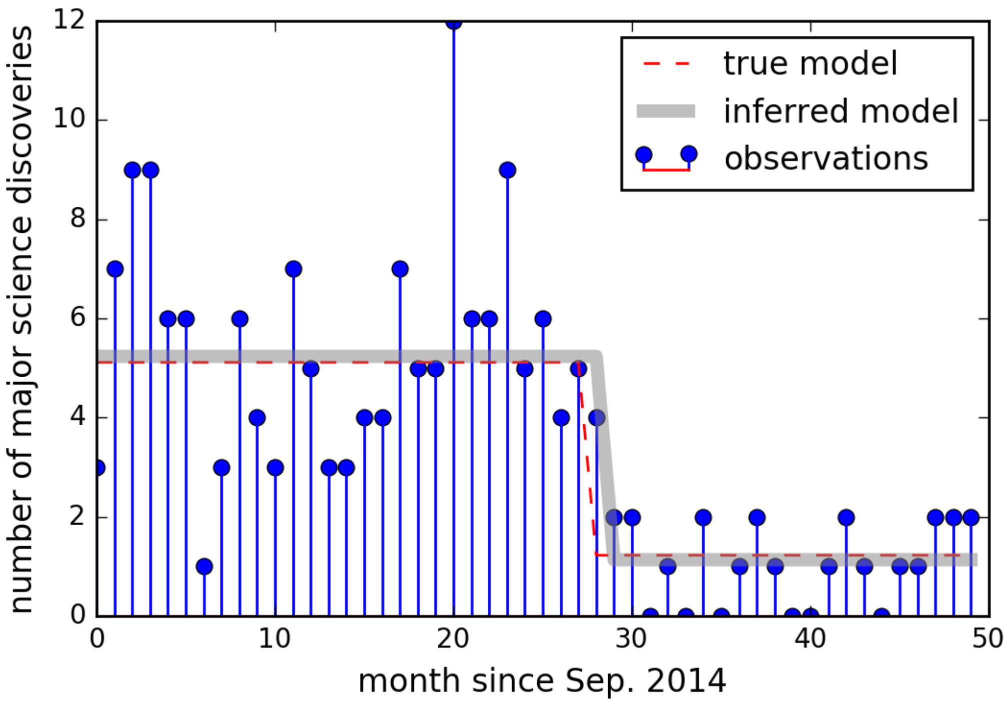
Gibbs sampling require full conditionals

$$\ln P(\lambda_1|\lambda_2, n, \boldsymbol{x}) = \ln Gamma(\lambda_1; a + \sum_{i=0}^{n-1} x_i, b + n)$$

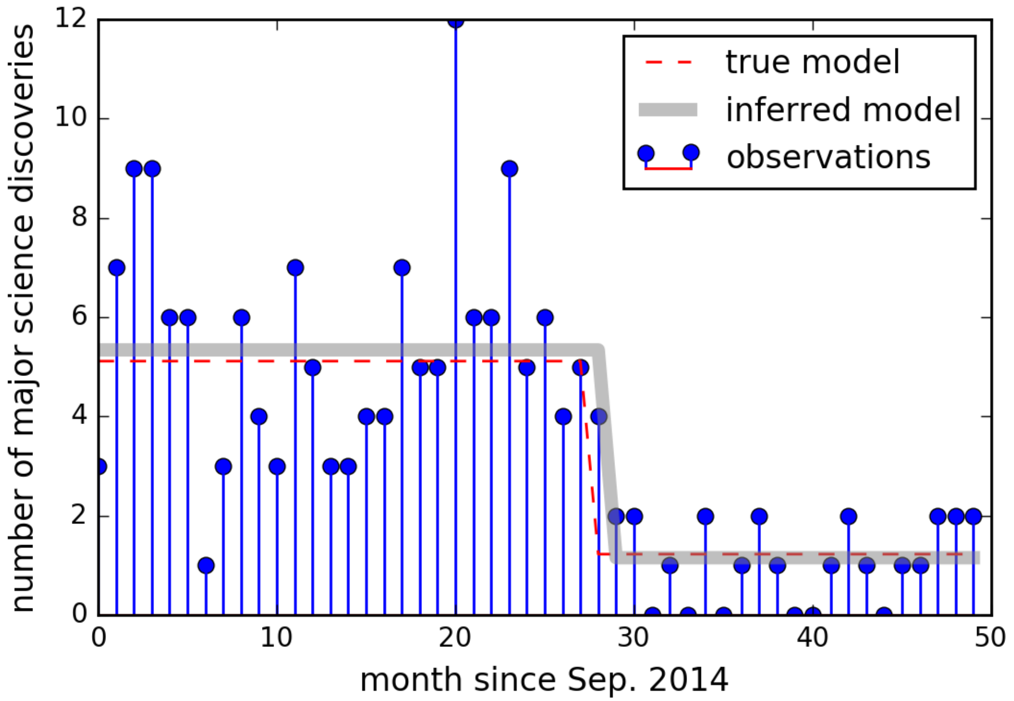
$$\ln P(\lambda_2|\lambda_1, n, \boldsymbol{x}) = \ln Gamma(\lambda_2; a + \sum_{i=n}^{N-1} x_i, b + N - n)$$

$$\ln P(n|\lambda_1, \lambda_2, \boldsymbol{x}) = mess(n|\lambda_1, \lambda_2, \boldsymbol{x})$$

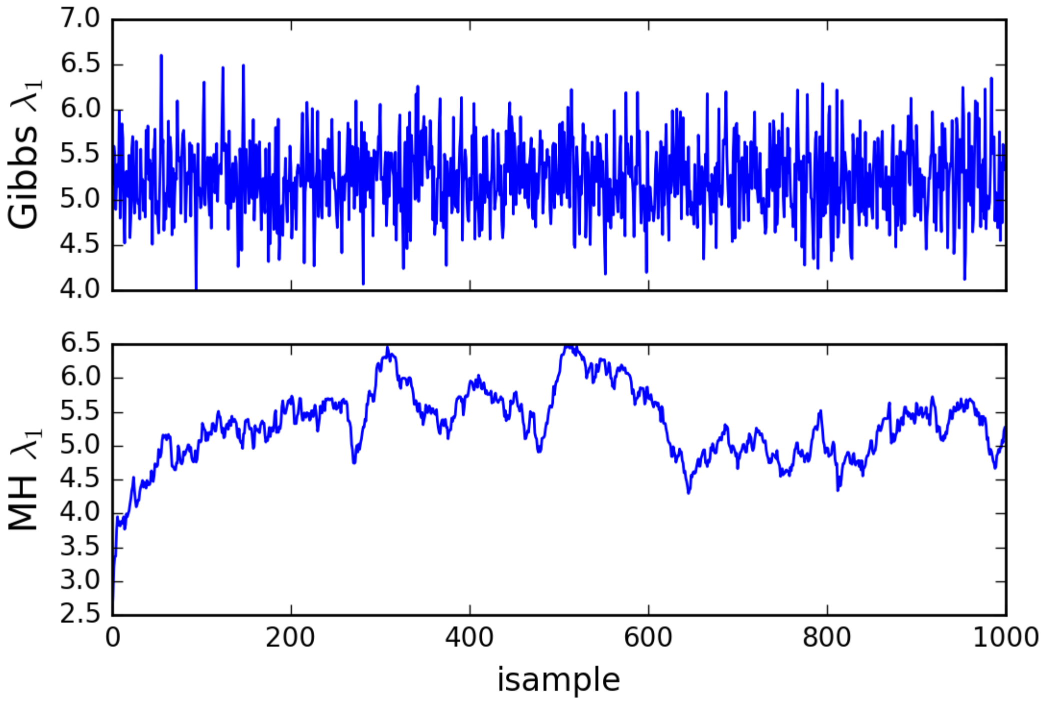
Q/How to sample this mess?!
A/ In general: Metropolis within Gibbs.
In this case: bruteforce $P(n|\lambda_1, \lambda_2, \boldsymbol{x}), \forall n = \{0, \dots, N-1\}$ because N is rather small.



Model Example: Train a Change-point Model with Bayesian Inference



Model sampled from Metropolis sampler



λ_1 samples from Gibbs and naïve Metropolis

Advanced Example: Train a Binary Restricted Boltzmann Machine on MNIST

Binary Restricted Boltzmann Machine (BRBM):

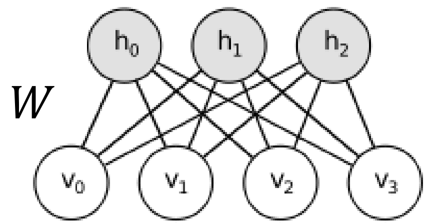
- A particular probability distribution of observables and model parameters
- The "machine" is specified by 2 real (shift) vectors and 1 real (weight) matrix
- The state of the "machine" is specified by 2 Binary vectors (hidden & visible)

See Dima's presentation for more detailed description of RBM: http://algorithm-interest-group.me/algorithm/Boltzmann-Machines-Dima-Kochkov/

$$P(\boldsymbol{v}, \boldsymbol{h}, W, \boldsymbol{a}, \boldsymbol{b}) = \frac{\exp[\boldsymbol{a}^T \boldsymbol{v} + \boldsymbol{b}^T \boldsymbol{h} + \boldsymbol{h}^T W \boldsymbol{v}]}{Z}$$

$$Z = \sum_{\boldsymbol{v},\boldsymbol{h}} \exp[\sum_{j=0}^{nvis-1} a_j v_j + \sum_{i=0}^{nhid-1} b_i h_i + \sum_{i,j} h_i W_{ij} v_j]$$

visualize



$nhid = 3$

$nvis = 4$

- In binary RBM, $\boldsymbol{v}, \boldsymbol{h}$ are vectors of 1s and 0s.

Advanced Example: Train a Binary Restricted Boltzmann Machine on MNIST

See Dima's presentation for more detailed description of RBM: http://algorithm-interest-group.me/algorithm/Boltzmann-Machines-Dima-Kochkov/

Binary Restricted Boltzmann Machine (BRBM):

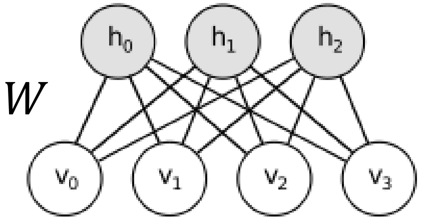
- A particular probability distribution of observables and model parameters
- The "machine" is specified by 2 real (shift) vectors and 1 real (weight) matrix
- The state of the "machine" is specified by 2 Binary vectors (hidden & visible)

$$P(\boldsymbol{v},\boldsymbol{h},W,\boldsymbol{a},\boldsymbol{b}) = \frac{\exp[\boldsymbol{a}^T\boldsymbol{v}+\boldsymbol{b}^T\boldsymbol{h}+\boldsymbol{h}^T W\boldsymbol{v}]}{Z}$$

$$Z = \sum_{\boldsymbol{v},\boldsymbol{h}} \exp[\sum_{j=0}^{nvis-1} a_j v_j + \sum_{i=0}^{nhid-1} b_i h_i + \sum_{i,j} h_i W_{ij} v_j]$$

visualize

W



$nhid = 3$

$nvis = 4$

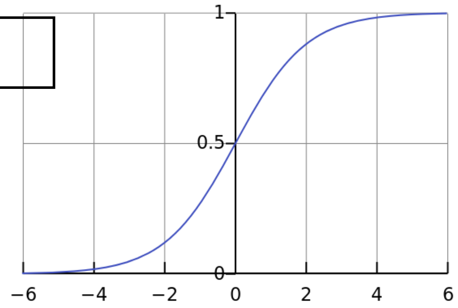
- In binary RBM, $\boldsymbol{v},\boldsymbol{h}$ are vectors of 1s and 0s.

Thus full conditionals are simple:

$$\frac{P(v_j=1|*)}{P(v_j=0|*)} = \frac{\exp[\boldsymbol{a}^T\boldsymbol{v}+\boldsymbol{b}^T\boldsymbol{h}+\boldsymbol{h}^T W\boldsymbol{v}]_{\mathrm{v_j=1}}}{\exp[\boldsymbol{a}^T\boldsymbol{v}+\boldsymbol{b}^T\boldsymbol{h}+\boldsymbol{h}^T W\boldsymbol{v}]_{\mathrm{v_j=0}}} = \exp\left[a_j + \sum_i h_i W_{ij}\right]$$

$$P(v_j=1|*) = \frac{P(h_i=1|*)}{P(h_i=1|*)+P(h_i=0|*)} = \frac{1}{1+\exp[-a_j-\sum_i h_i W_{ij}]} = sigmoid(a_j+\sum_i h_i W_{ij})$$

Notice no matrix element among v_j (restricted), thus: $\boxed{P(\boldsymbol{v}=1|*) = sigmoid(\boldsymbol{a}+W^T\boldsymbol{h})}$



That is: we can sample binary RBM efficiently with *block Gibbs sampling*!

Advanced Example: Train a Binary Restricted Boltzmann Machine on MNIST

Q/ How to “train” a BRBM?

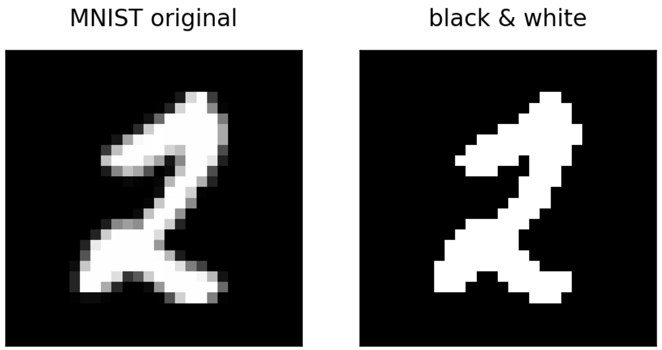
Q1/ What is the outcome/goal of “training”?

Q2/ What are the inputs in a “training”?

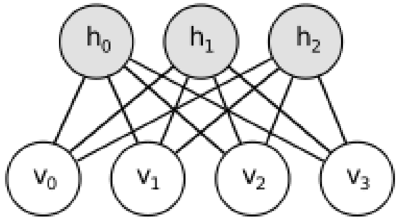
Q3/ What does it mean to “train”?

Q4/ What changes in the “training”?

MNIST database:
70,000 handwritten digits from 0 to 9



Each picture has 28×28 gray scale pixels {0,1,…,255}. For input into the BRBM, scale to [0,1.0) and cutoff at 0.5.



nvis = 28×28 = 784

Advanced Example: Train a Binary Restricted Boltzmann Machine on MNIST

Q/ How to "train" a BRBM?

Q1/ What is the outcome/goal of "training"?
A1/ A joint probability distribution of 784 Bernoulli random variables, which favors configurations that look like digits. i.e. want $P(\boldsymbol{v}|*)$ that represents data.

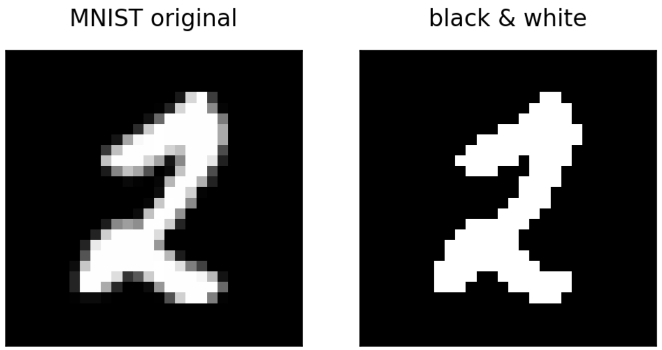
Q2/ What are the inputs in a "training"?
A2/ $\boldsymbol{v}_s$, s=1,2,…,ndata. Each $\boldsymbol{v}_s$ is a vector 784 0s and 1s.

Q3/ What does it mean to "train"?
A3/ Increase the probability of $P(\boldsymbol{v}_s|*)$.

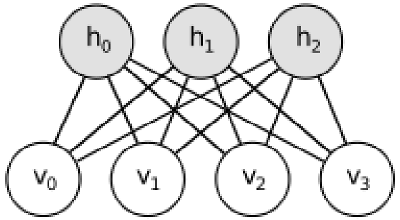
Q4/ What changes in the "training"?
A4/ The "machine". Specifically: $\{\mathbf{a}, \boldsymbol{b}, W\}$

A/ Increase $P(\boldsymbol{v}_s|*), \forall s$ by changing $\{\mathbf{a}, \boldsymbol{b}, W\}$.

MNIST database:
70,000 handwritten digits from 0 to 9



Each picture has 28×28 gray scale pixels {0,1,…,255}. For input into the BRBM, scale to [0,1.0) and cutoff at 0.5.



nvis = 28×28 = 784

Advanced Example: Train a Binary Restricted Boltzmann Machine on MNIST

- Gradient of cost function (ref: http://deeplearning.net/tutorial/rbm.html)

$$\frac{\partial \ln P}{\partial W_{ij}} = < h_i v_j >_{data} - < h_i v_j >_{model}$$

$$P(\boldsymbol{v} = 1 | *) = sigmoid(\boldsymbol{a} + W^T \boldsymbol{h})$$

$$P(\boldsymbol{h} = 1 | *) = sigmoid(\boldsymbol{b} + W\, \boldsymbol{v})$$

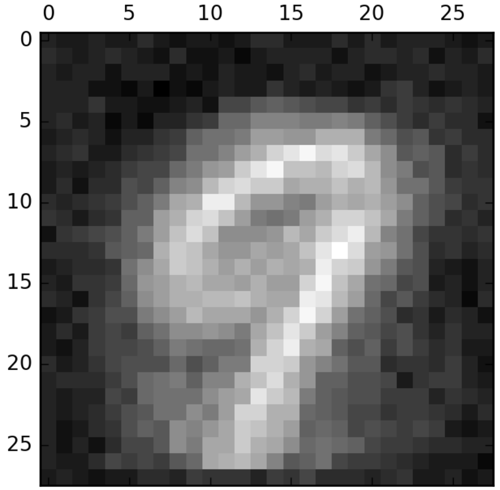
```python
def get_vis(h,W,a):
    return map(int, np.random.rand(len(a)) < sigmoid(np.dot(W.T,h)+a) )
# end def
def get_hid(v,W,b):
    return map(int, np.random.rand(len(b)) < sigmoid(np.dot(W,v)+b) )
# end def
```

- Training procedure: Contrastive Divergence (a.k.a. shitty steepest decent)

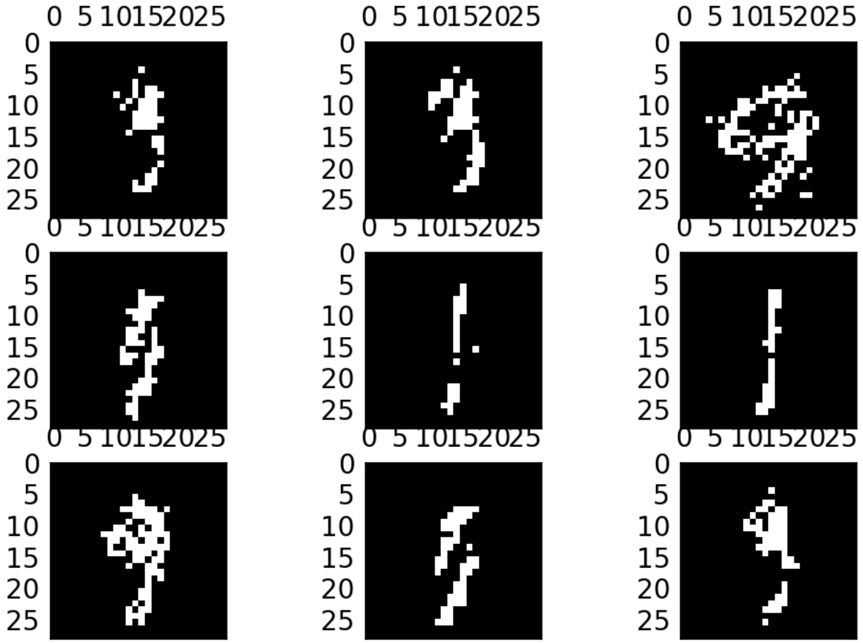
G.E. Hinton, A Practical Guide to Training Restricted
Boltzmann Machines, *Neural Networks: Tricks of the Trade*, vol. 7700, pp 599-619, 2010.

Advanced Example: Train a Binary Restricted Boltzmann Machine on MNIST

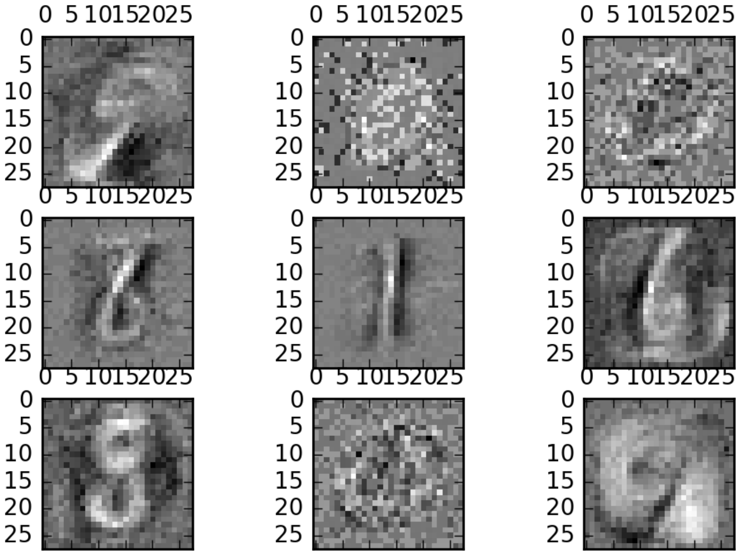
Shift vector for visible units $\boldsymbol{a}$



BRBM samples after training

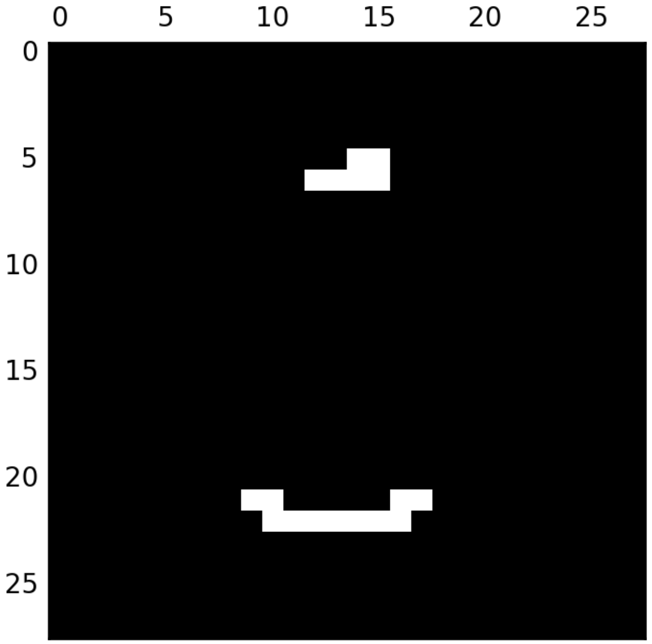


Rows of weight matrix W
(ordered by shift vector for hidden units $\boldsymbol{b}$)



Advanced Example: Train a Binary Restricted Boltzmann Machine on MNIST

Q/ What number is this?



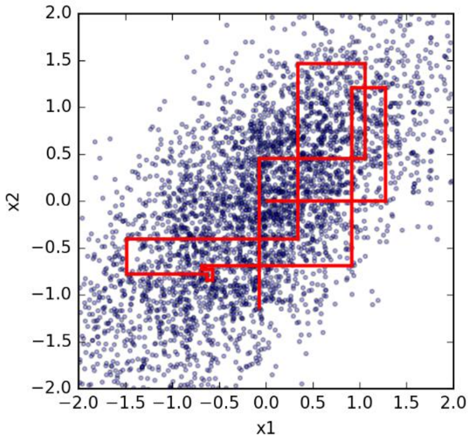
Conclusions:

Pros:

- The *Gibbs sampling* technique draws samples from a multivariate probability distribution by sampling the full conditional of each variable in turn.
- Independent variables can be sampled simultaneously, making *block Gibbs sampling* highly efficient for certain distributions.

Cons:

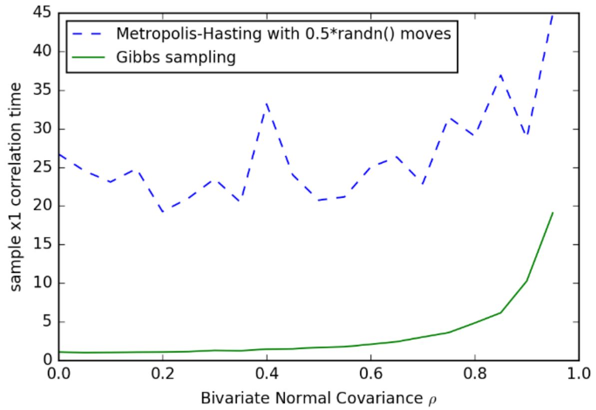
- Calculating full conditionals may be intractable and error prone
- Fails when random variables are nearly perfectly correlated



$$P(\boldsymbol{v} = 1|\, *) = sigmoid(\boldsymbol{a} + W^T \boldsymbol{h})$$

$$P(\boldsymbol{h} = 1|\, *) = sigmoid(\boldsymbol{b} + W\, \boldsymbol{v})$$

```
def get_vis(h,W,a):
    return map(int, np.random.rand(len(a)) < sigmoid(np.dot(W.T,h)+a) )
# end def
def get_hid(v,W,b):
    return map(int, np.random.rand(len(b)) < sigmoid(np.dot(W,v)+b) )
# end def
```



References

Bivariate Normal Distribution:

- MCMC: The Gibbs Sampler, The Clever Machine
- Bayesian Inference: Metropolis-Hasting Sampling, Ilker Yildirim

Change-point Model:

- Bayesian Inference: Gibbs Sampling, Ilker Yildirim

Restricted Boltzmann Machine:

- A Practical Guide to Training Restricted Boltzmann Machines, Geoffrey E. Hinton
- deeplearning.net
- Introduction to Restricted Boltzmann Machines, Edwin Chen