MARKOV RANDOM FIELDS AND IMAGE RESTORATION

Alec Bunnell, Anne Grosse, Jialun Luo, Sam Spaeth

February 24, 2016

Carleton College
Northfield, MN

INTRODUCTION

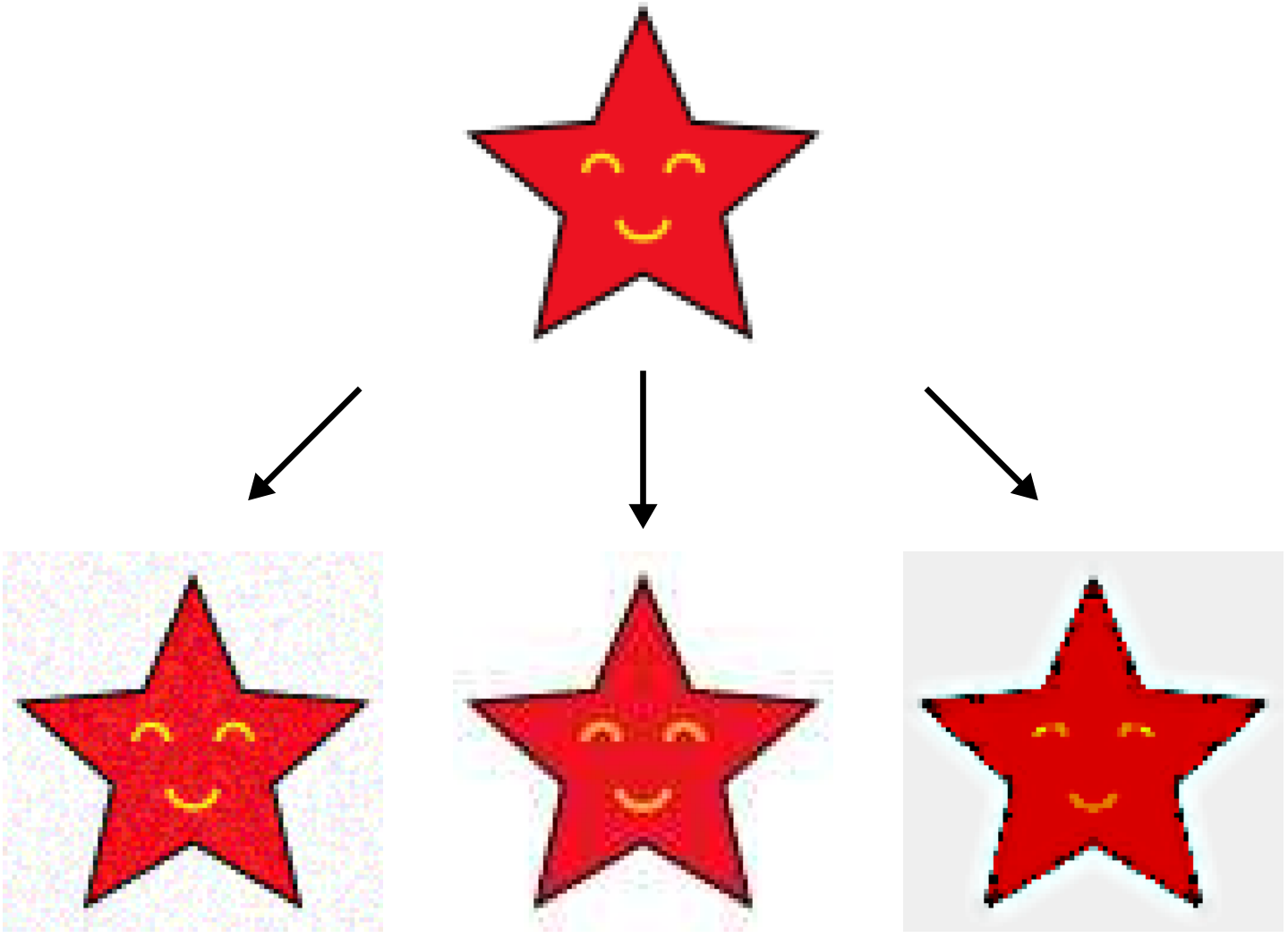
MOTIVATION

Goal: Recover an image in its original quality, given only a noisy version of it.

MOTIVATION

Goal: Recover an image in its original quality, given only a noisy version of it.

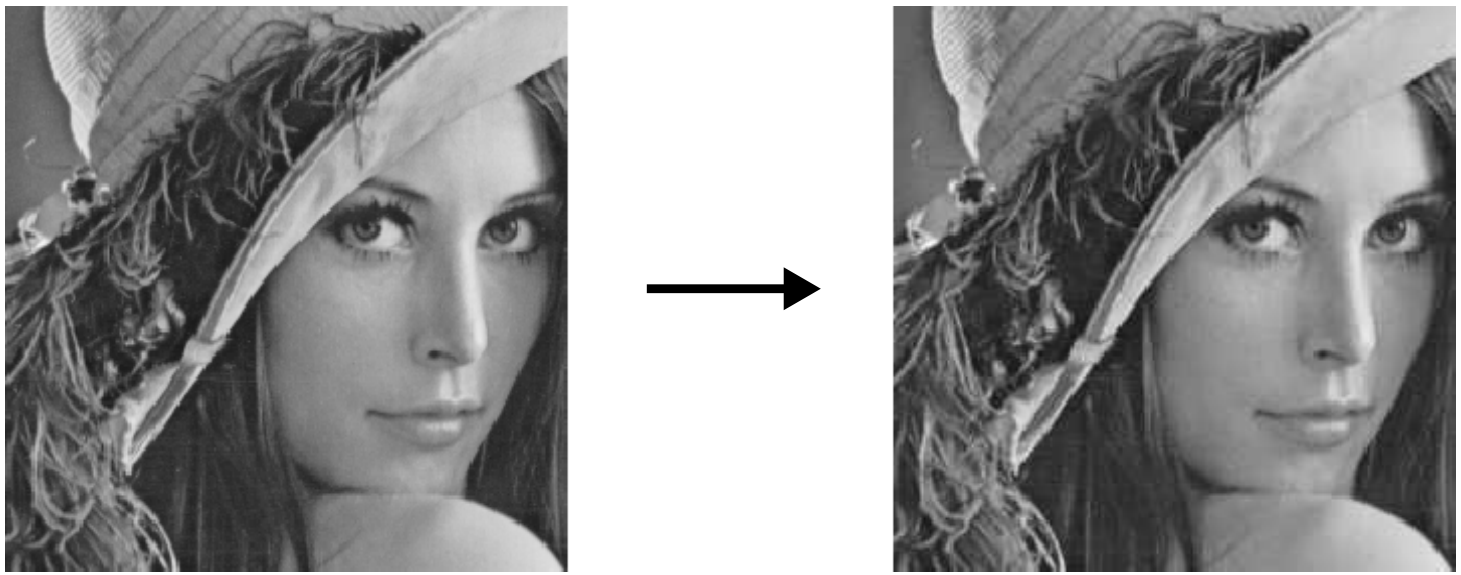
DIFFERENT NOISE MODELS



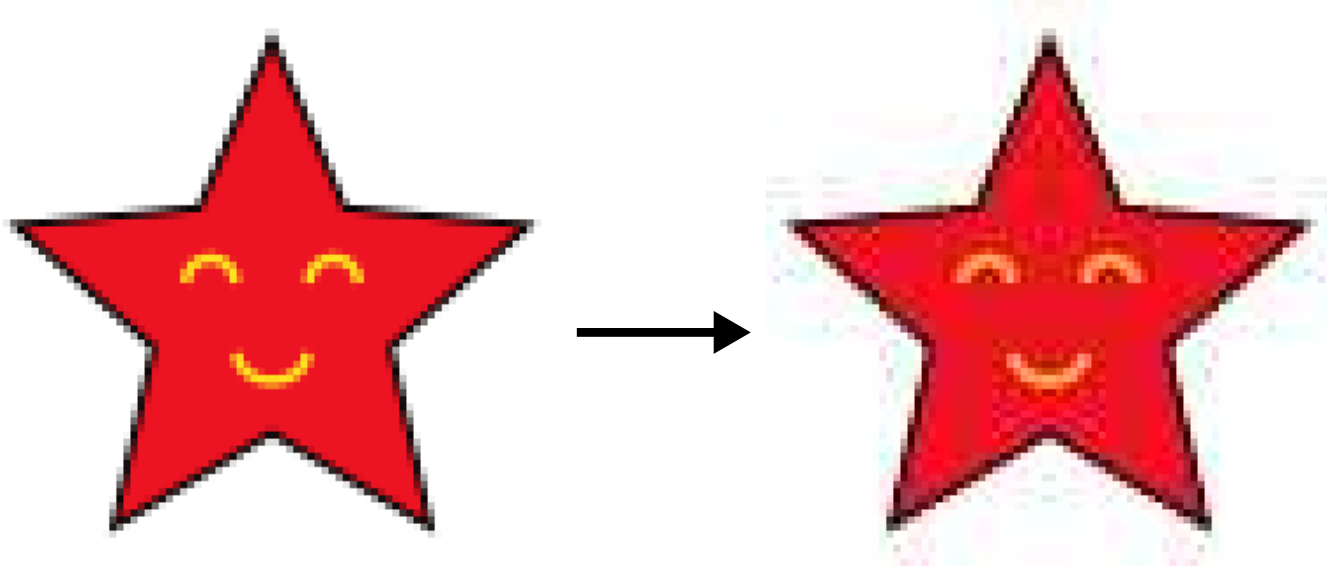
ONE NOISE MODEL TO RULE THEM ALL

JPEG:

- Uses lossy compression to reduce filesize, variable quality level
- Designed to be relatively unnoticeable on photos (i.e., of real-life subjects)



JPEG compression is significantly more noticeable on cartoon-like images and text.



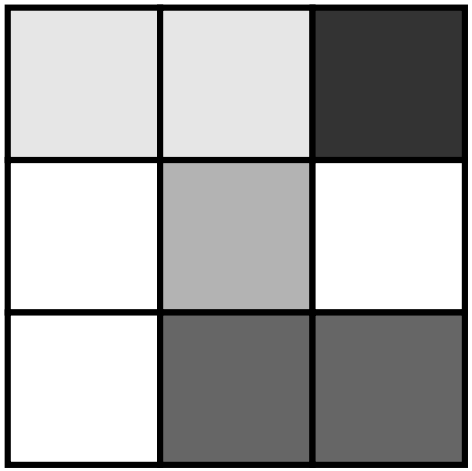
OVERVIEW

Goal: Undo the error introduced by JPEG compression

- Non-invertible compression
- Best guess for the original image
- Iteratively propose candidates, gradually improving towards our final recovered image

Image:

An image x is a 2-D grid of pixels, each of which has value $x_s \in [0, 1]$ for $s \in V$, where V is the set of pixels.



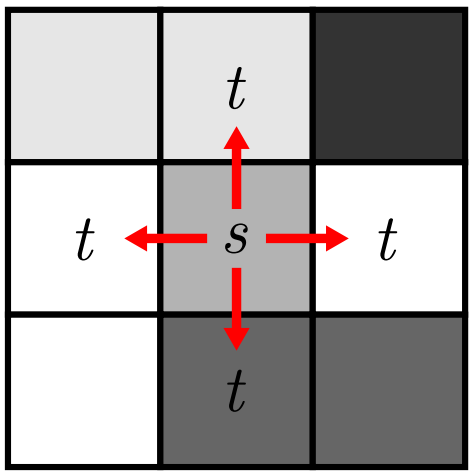
$=$

0.9	0.9	0.2
1.0	0.7	1.0
1.0	0.4	0.4

For now, we'll disregard color and work only in the grayscale setting.

Neighbors:

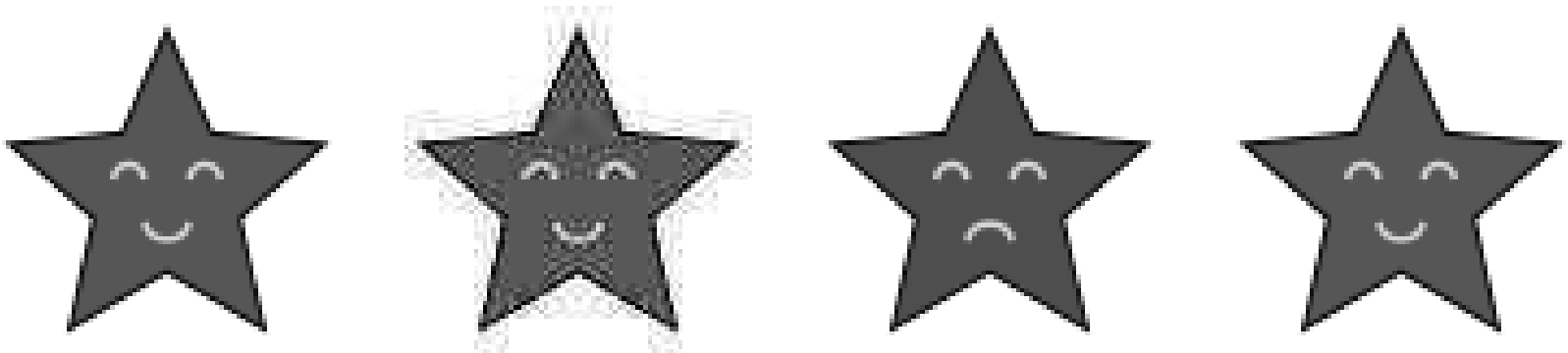
We use the notation $t \sim s$ to say that t is a neighboring pixel of s.



For our purposes, we'll consider a pixel's neighbors to be the four pixels adjacent to it.

Notation:

y^* = the original, full quality image
y = the given, degraded version of the original
x = an arbitrary proposal image (a candidate for x^*)
x^* = optimal candidate; our best attainable estimate of the original

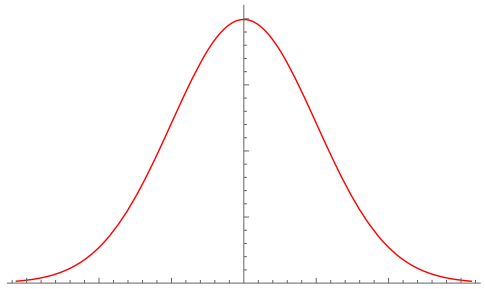


y^* y x x^*

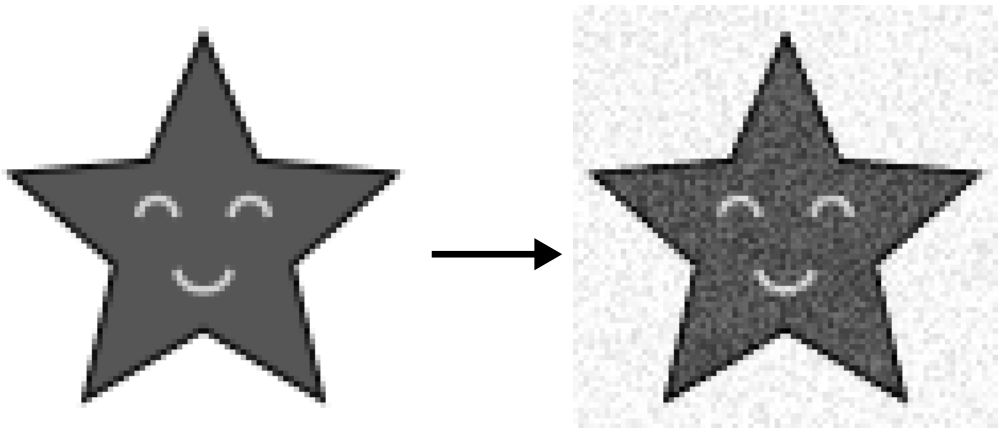
Ideally, we would like our algorithm to produce $x^* = y^*$.

A SMALL SIMPLIFICATION

We'll start by considering a simpler noise model: i.i.d. additive Gaussian noise.



Under this model, each pixel has an independent amount of normally-distributed jitter applied to it.



THE ENERGY FUNCTION

Energy Function:

$H(x)$ quantifies the "quality" of image x.

- $H(x)$ should be large (high energy) if x is a bad candidate.
- $H(x)$ should be small (low energy) if x is more ideal.

Energy Function:

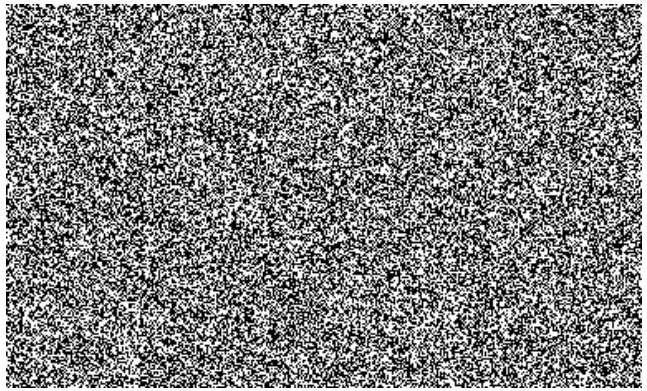
$H(x)$ quantifies the "quality" of image x.

- $H(x)$ should be large (high energy) if x is a bad candidate.
- $H(x)$ should be small (low energy) if x is more ideal.

Original Low Energy High Energy

FAVORING SMOOTHNESS

What does a general image look like?



FAVORING SMOOTHNESS

We define F to sum over a candidate image, comparing each pixel to its four neighboring pixels:

$$F(x) = \sum_s f(s),$$

with

$$f(s) = \sum_{t \sim s} (x_s - x_t)^2$$

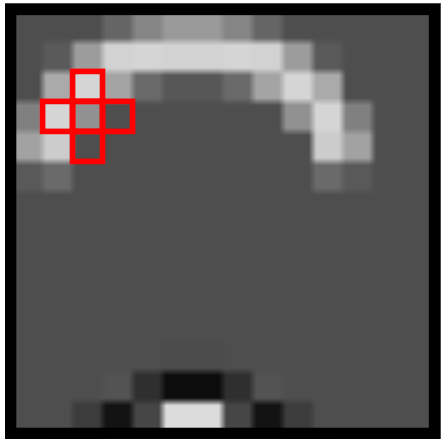
FAVORING SMOOTHNESS

We define F to sum over a candidate image, comparing each pixel to its four neighboring pixels:

$$F(x) = \sum_s f(s),$$

with

$$f(s) = \sum_{t \sim s} (x_s - x_t)^2$$



x

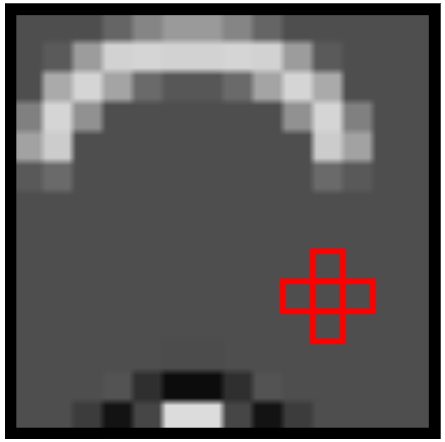
FAVORING SMOOTHNESS

We define F to sum over a candidate image, comparing each pixel to its four neighboring pixels:

$$F(x) = \sum_s f(s),$$

with

$$f(s) = \sum_{t \sim s} (x_s - x_t)^2$$



x

PERHAPS A TAD TOO SMOOTH...

If $H(x) = F(x)$, one minimizer of the function looks like:

Any solid-color image would be an optimal (zero-energy) candidate for this choice of H.

INCORPORATING THE GIVEN DATA

Solution: Penalize candidates that deviate far from the observed image.

INCORPORATING THE GIVEN DATA

Solution: Penalize candidates that deviate far from the observed image. Define:

$$D(x) = \sum_s d(s),$$

with

$$d(s) = (x_s - y_s)^2$$

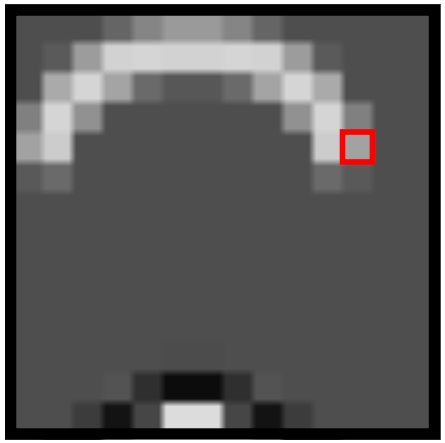
INCORPORATING THE GIVEN DATA

Solution: Penalize candidates that deviate far from the observed image. Define:

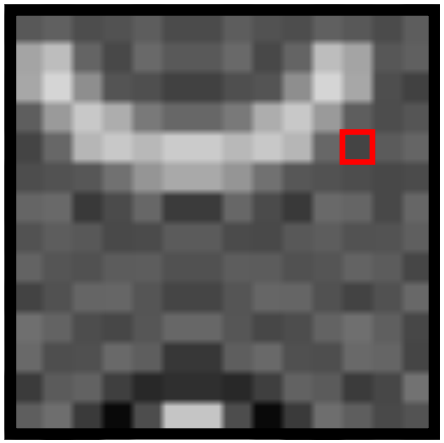
$$D(x) = \sum_s d(s),$$

with

$$d(s) = (x_s - y_s)^2$$



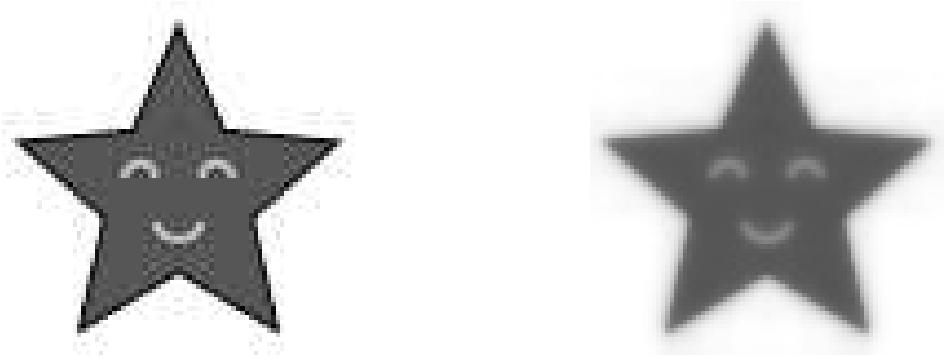
x



y

IGNORING MEANINGFUL CONTRAST

Using $H(x) = F(x) + D(x)$ would produce something more like:

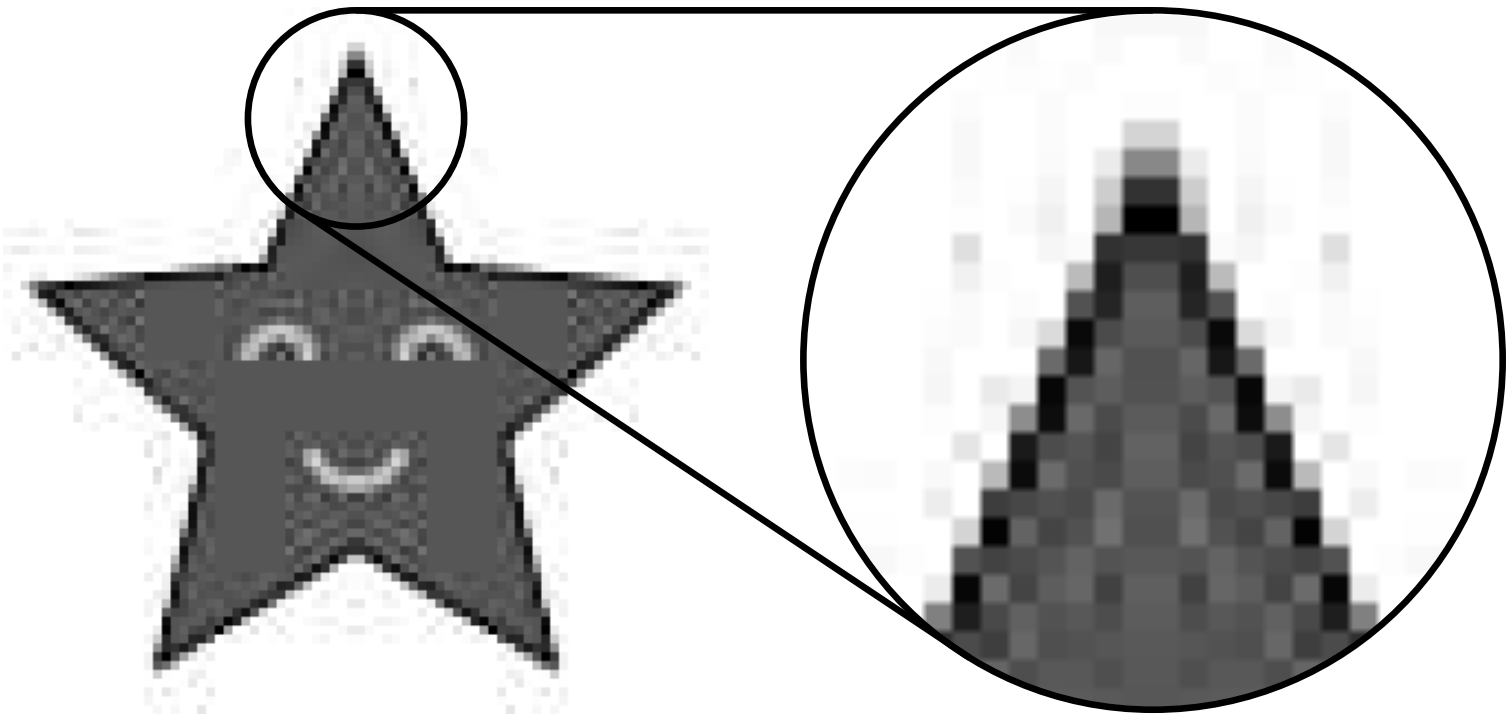


(We can control the tradeoff between F and D by scaling $D(x)$ by some parameter θ.)

We have balanced the over-smoothing with the noisy image, but there are still points in the image where no blurring should occur.

But not all contrast in an image is JPEG noise.

WHAT IS MEANINGFUL CONTRAST?



PRESERVING MEANINGFUL CONTRAST

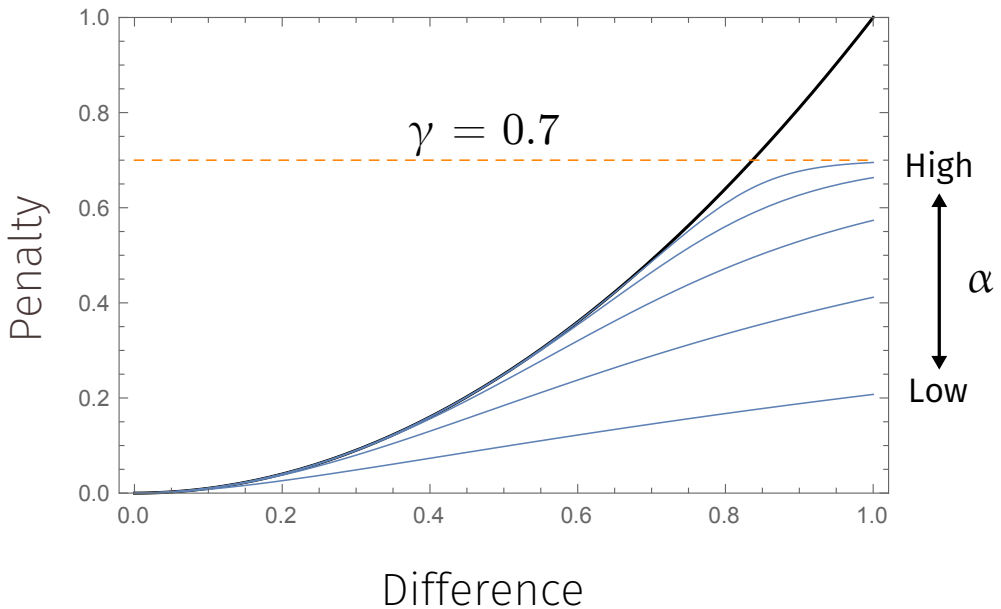
Solution: Alter $F(x) = \sum_s f(s)$ to tolerate great neighbor pixel differences.

PRESERVING MEANINGFUL CONTRAST

Solution: Alter $F(x) = \sum_s f(s)$ to tolerate great neighbor pixel differences.

$$f(s) = \sum_{t \sim s} \left(|x_s - x_t|^{-2\alpha} + \gamma^{-\alpha} \right)^{-1/\alpha}$$

- γ is the maximum penalty
- α determines how quickly the penalty approaches the limit



MEANINGFUL CONTRAST PRESERVATION IN ACTION

The results are sensitive to our choice of γ and α:

MOVING FORWARD

Now that we have a satisfactory energy function, our goal is to minimize it.

Instead of directly searching for a minimizer, we'll use it to model the probability that a given candidate x is the original image.

THE BAYESIAN PARADIGM

- Consider an arbitrary candidate image x.
- We would like to find $P(x \text{ original} \mid y \text{ observed})$ to assess how good of a guess x is for the original image.

MOTIVATION

- Consider an arbitrary candidate image x.
- We would like to find $P(x \text{ original} \mid y \text{ observed})$ to assess how good of a guess x is for the original image.
- Using our noise model, $P(y \text{ observed} \mid x \text{ original})$ is straightforward.
- Unfortunately, conditional probabilities aren't symmetric...

BAYES' RULE

Bayes' rule uses properties of conditional probability to provide a means for "inverting" a conditional probability $P(A \mid B) \rightarrow P(B \mid A)$.

$$P(B \mid A) = \frac{P(A \cap B)}{P(A)} = \frac{P(A \mid B)P(B)}{P(A)}$$

BAYES' RULE APPLIED TO IMAGES

$$P(x \text{ original} \mid y \text{ observed}) = \frac{P(y \text{ observed} \mid x \text{ original}) P(x \text{ original})}{P(y \text{ observed})}$$

Since y is a fixed image (our observed data), we have

$$\boldsymbol{P(x \text{ original} \mid y \text{ observed}) \propto P(x \text{ original}) P(y \text{ observed} \mid x \text{ original})}$$

$$\underbrace{P(x \text{ original} \mid y \text{ observed})}_{\text{posterior}} \propto \underbrace{P(x \text{ original})}_{\text{prior}} \underbrace{P(y \text{ observed} \mid x \text{ original})}_{\text{likelihood}}$$

Main Idea:

- Begin with some prior expectations about what you think a general image should look like.
- Take into account the likelihood of obtaining the given degraded image assuming that things started from our candidate.
- Yield an updated, posterior probability distribution describing our candidate.

THE ENERGY FUNCTION STRIKES BACK

- Recall: We would like to find a candidate image x such that it minimizes our energy function $H(x)$.
- We'll construct a probability distribution in which low energy candidates are the most likely.
- Use $\Pi(x) \propto e^{-H(x)}$.

BACK TO THE BAYES-ICS

$$\begin{aligned}\Pi(x) &\propto e^{-H(x)} \\ &\propto e^{-F(x)-D(x)} \\ &\propto e^{-F(x)}\, e^{-D(x)}\end{aligned}$$

Observe that

- $F(x)$ describes the energy *within* the candidate, based on our prior expectations of what makes an image "ideal" (prior)
- $D(x)$ disallows the candidate from straying too far from the observed data (likelihood)
- $\Pi(x)$ will favor candidates that satisfy both of these conditions in an optimally balanced way (posterior)

OUR POSTERIOR DISTRIBUTION

We thus have the following model for the probability that a candidate x is our source image, given only the degraded image y:

OUR POSTERIOR DISTRIBUTION

We thus have the following model for the probability that a candidate x is our source image, given only the degraded image y:

$$\begin{aligned}\Pi(x) &\propto e^{-H(x)} \\ &\propto e^{-\sum_s \left(\sum_{t \sim s} \left(|x_s - x_t|^{-2\alpha} + \gamma^{-\alpha} \right)^{-1/\alpha} + \theta (x_s - y_s)^2 \right)}\end{aligned}$$

SAMPLING FROM THE POSTERIOR

We have now constructed a distribution on the space of images such that “better” images are considered more likely.

If we could sample from Π, we’d get these “better” images most often.

Sampling images according to Π would be a fine way of proposing candidates!

“Houston, we have a problem”

To sample from Π directly, we must enumerate the space of all possible images and find the constant of proportionality.

“Houston, we have a problem”

To sample from Π directly, we must enumerate the space of all possible images and find the constant of proportionality.

- For an $R \times C$ image stored with k discrete gray levels, there are k^{RC} unique image possibilities.

"HOUSTON, WE HAVE A PROBLEM"

To sample from Π directly, we must enumerate the space of all possible images and find the constant of proportionality.

- For an $R \times C$ image stored with k discrete gray levels, there are k^{RC} unique image possibilities.
- At 256 gray-levels (standard), even a 6×6 image has $256^{6 \times 6} \sim 10^{86}$ possibilities.

"HOUSTON, WE HAVE A PROBLEM"

To sample from Π directly, we must enumerate the space of all possible images and find the constant of proportionality.

- For an $R \times C$ image stored with k discrete gray levels, there are k^{RC} unique image possibilities.
- At 256 gray-levels (standard), even a 6×6 image has $256^{6\times 6} \sim 10^{86}$ possibilities.
- As a comparison, the estimated number of particles in the known universe is $\sim 10^{80}$.

"HOUSTON, WE HAVE A PROBLEM"

To sample from Π directly, we must enumerate the space of all possible images and find the constant of proportionality.

- For an $R \times C$ image stored with k discrete gray levels, there are k^{RC} unique image possibilities.
- At 256 gray-levels (standard), even a 6×6 image has $256^{6\times 6} \sim 10^{86}$ possibilities.
- As a comparison, the estimated number of particles in the known universe is $\sim 10^{80}$.

So sampling from Π will be a bit of a challenge...

THE GIBBS SAMPLER

THE GIBBS SAMPLER

- Treat each pixel's value x_s as a random variable X_s.
- A random image is a collection of these random pixels, $X = (X_s)_{s \in V}$.
- The Gibbs Sampler allows sampling from a joint distribution of random variables.
- The distribution of a random image X is simply the joint distribution of all its pixels.

Instead of proposing a completely new candidate at each iteration, propose a new image by changing one pixel at a time.

Steps:

- Randomly choose a pixel s to update.
- Find the distribution of $X_s \mid X_t, \forall t \neq s$.
- Randomly select a new value for x_s according to this distribution.
- Repeat.

ONE MORE PROBLEM

The conditional distribution of $X_s \mid X_t, \forall t \neq s$ is not immediately obvious...

MARKOV RANDOM FIELDS

Definition

A Markov Chain is a sequence of random variables $X_1, X_2, \ldots$ that satisfies

$$P(X_{n+1} = x_{n+1} \mid X_1 = x_1, \ldots, X_n = x_n) = P(X_{n+1} = x_{n+1} \mid X_n = x_n)$$

for all steps n.

Definition

A Markov Chain is a sequence of random variables $X_1, X_2, \ldots$ that satisfies

$$P(X_{n+1} = x_{n+1} \mid X_1 = x_1, \ldots, X_n = x_n) = P(X_{n+1} = x_{n+1} \mid X_n = x_n)$$

for all steps n. **At each point in the chain, the next step is dependent only on the current step, regardless of all other steps.**

Definition

A Markov Chain is a sequence of random variables $X_1, X_2, \ldots$ that satisfies

$$P(X_{n+1} = x_{n+1} \mid X_1 = x_1, \ldots, X_n = x_n) = P(X_{n+1} = x_{n+1} \mid X_n = x_n)$$

for all steps n. **At each point in the chain, the next step is dependent only on the current step, regardless of all other steps.**

This is known as the **Markov property** for Markov chains.

Definition

A Markov Chain is a sequence of random variables $X_1, X_2, \ldots$ that satisfies

$$P(X_{n+1} = x_{n+1} \mid X_1 = x_1, \ldots, X_n = x_n) = P(X_{n+1} = x_{n+1} \mid X_n = x_n)$$

for all steps n. **At each point in the chain, the next step is dependent only on the current step, regardless of all other steps.**

This is known as the **Markov property** for Markov chains.

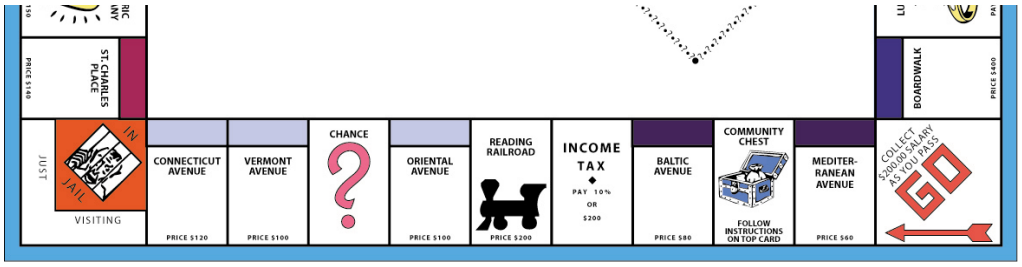
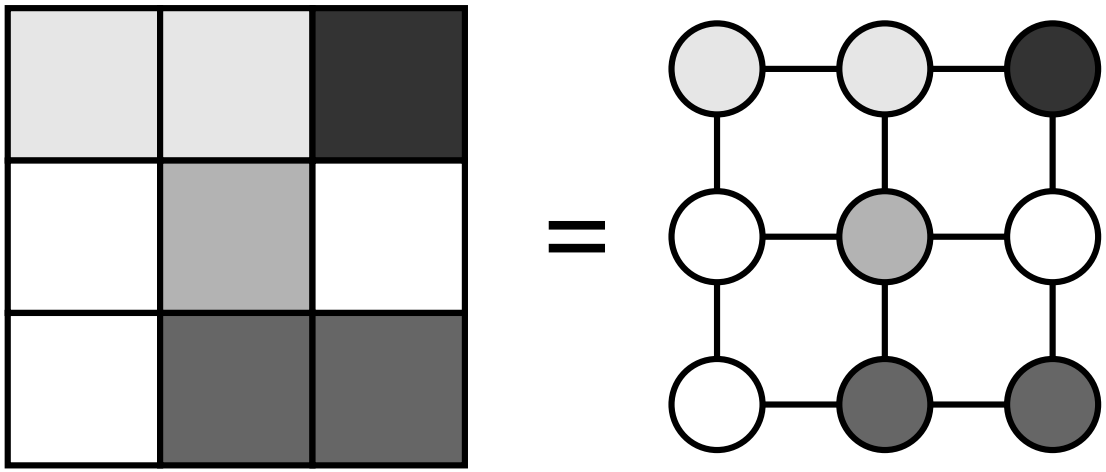


IMAGE AS A GRAPH

An image can be represented by a graph with vertices V and edges connecting adjacent pixels.



Recall that the value of pixel s (now vertex s) is a random variable X_s.

A Markov Random Field is a graph that satisfies

$$P(X_s = x_s \mid X_t = x_t, \forall t \neq s) = P(X_s = x_s \mid X_t = x_t, \forall t \sim s)$$

for all configurations X.

This is analogous to the Markov Property defined for Markov Chains.

The probability distribution for one vertex given the entire configuration requires us to look no further than its neighbors.

Theorem (Hammersley–Clifford):

A probability distribution with respect to a graph structure imposes a Markov Random Field if and only if the distribution is Gibbsian.

Definition:

A distribution Π is **Gibbsian** if and only if $\Pi(x) \propto e^{-H(x)}$, where:

- $H(x)$ is a strictly positive function
- $H(x) = \sum_s h(s)$, where $h(s)$ depends only on $t \sim s$.

Definition:

A distribution Π is **Gibbsian** if and only if $\Pi(x) \propto e^{-H(x)}$, where:

- $H(x)$ is a strictly positive function
- $H(x) = \sum_s h(s)$, where $h(s)$ depends only on $t \sim s$.

Our posterior distribution:

$$\Pi(x) \propto e^{-H(x)} = e^{-\sum_s h(s)}$$

where $h(s) = f(s) + d(s)$.

Thus, our energy function imposes a Markov Random Field on an arbitrary image

Since we have a Markov Random Field, the Markov Property must hold for our choice of Π. Specifically, we have that

$$\begin{aligned} P(X_s = x_s \mid X_t = x_t, \forall t \neq s) &= P(X_s = x_s \mid X_t = x_t, \forall t \sim s) \\ &\propto e^{-h(s)} \end{aligned}$$

Thus, to implement the Gibbs sampler, only calculations using the neighbors of a pixel s are required to update that pixel.

SAMPLING VS RECOVERING

- Due to the Gibbs sampler, we can now *sample* images from Π.
- However, we don't just want a sample of images which are just "likely" to have low energy; we want to produce *one* image that has the *lowest* energy.
- This is called the *maximum a posteriori* (MAP) estimate.

SIMULATED ANNEALING

ANNEALING

SIMULATED ANNEALING

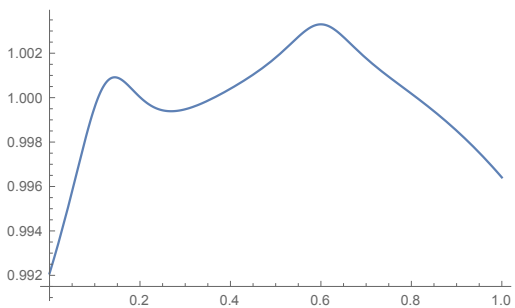
Introduce parameter β to our posterior distribution: $e^{-\beta H(x)}$.

By varying β, we can control how "peaked" our distribution is.

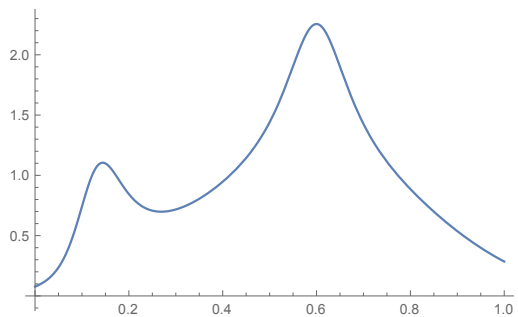
SIMULATED ANNEALING

Introduce parameter β to our posterior distribution: $e^{-\beta H(x)}$.

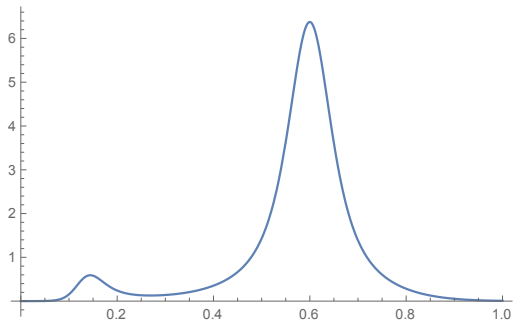
By varying β, we can control how "peaked" our distribution is.



$\beta = 1$



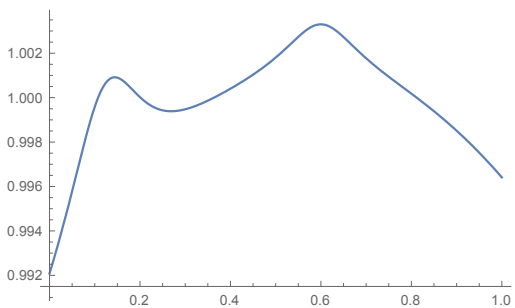
$\beta = 300$



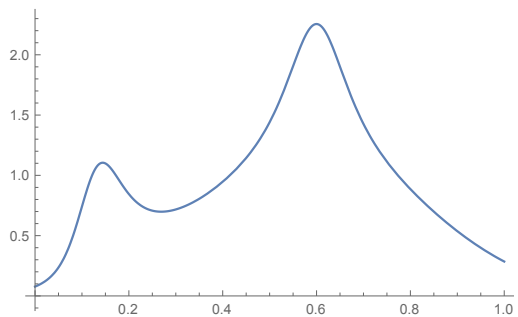
$\beta = 1000$

Introduce parameter β to our posterior distribution: $e^{-\beta H(x)}$.

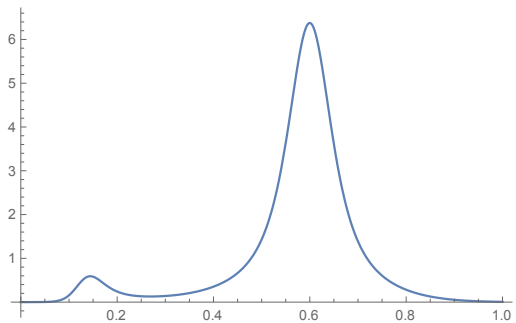
By varying β, we can control how "peaked" our distribution is.



$\beta = 1$



$\beta = 300$



$\beta = 1000$

We'll call this parameter inverse temperature.

- High "temperature" (low β) = lots of "jumping around"
- Low "temperature" (high β) = not as much "movement"

Cooling Schedule

A sequence of inverse temperatures $\beta_1, \ldots, \beta_n, \ldots$ modifying the posterior distribution Π is called a cooling schedule.

So at nth step, the posterior distribution is given by $\Pi(x) \propto e^{-\beta_n H(x)}$.

Our Gibbs Sampler conditional likewise becomes proportional to $e^{-\beta_n h(s)}$.

Theorem (Geman & Geman)

If a cooling schedule $\beta_1, \ldots, \beta_n, \ldots$ satisfies

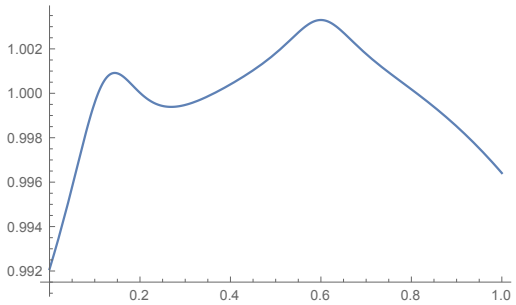
$$\beta_n \leq \frac{1}{RC(\gamma + \theta)} \log n$$

then we will eventually sample the mode of the posterior distribution (or the minimum-energy image) with probability 1.

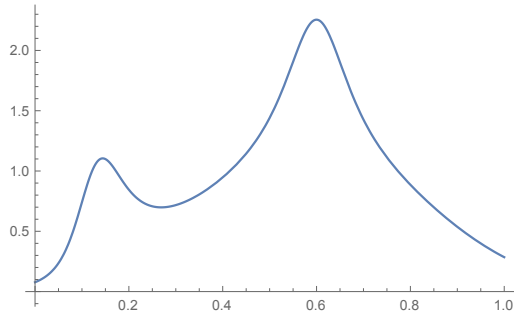
IMPLEMENTATION AND ADAPTATIONS

MODIFIED COOLING SCHEDULE

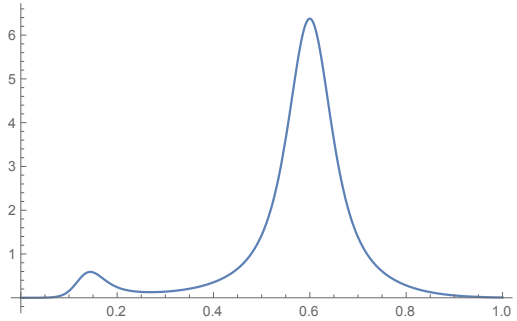
MODIFIED COOLING SCHEDULE



$\beta = 1$

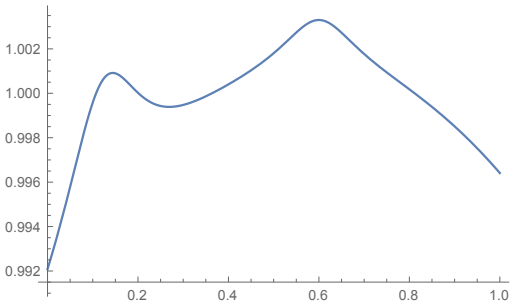


$\beta = 300$

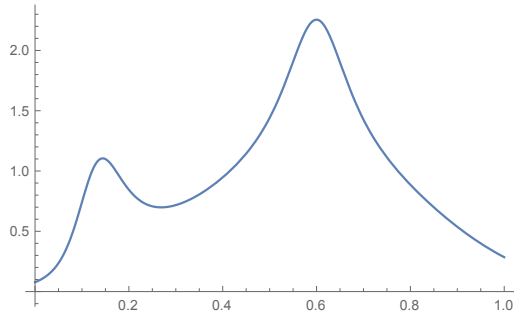


$\beta = 1000$

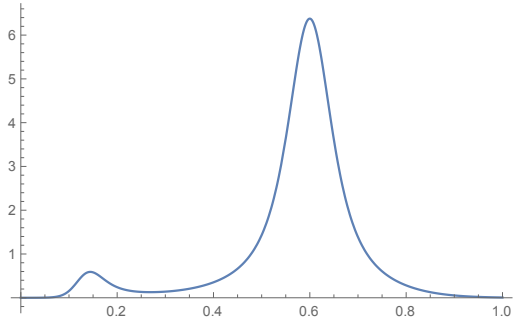
MODIFIED COOLING SCHEDULE



$\beta = 1$



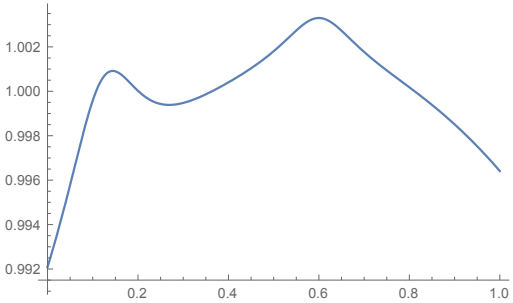
$\beta = 300$



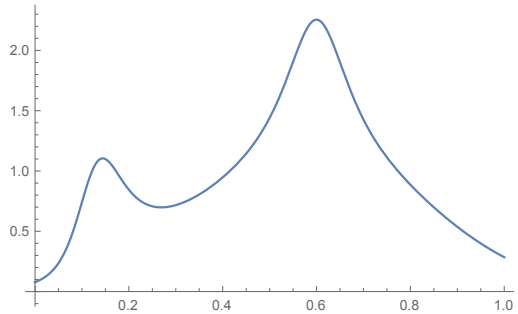
$\beta = 1000$

- Sticking to the exact bound given by Geman–Geman is incredibly slow.

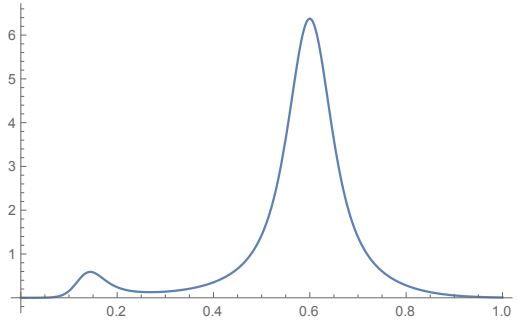
MODIFIED COOLING SCHEDULE



$\beta = 1$



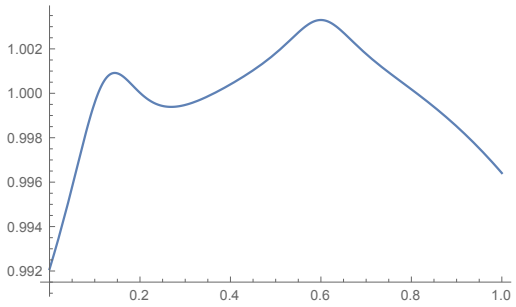
$\beta = 300$



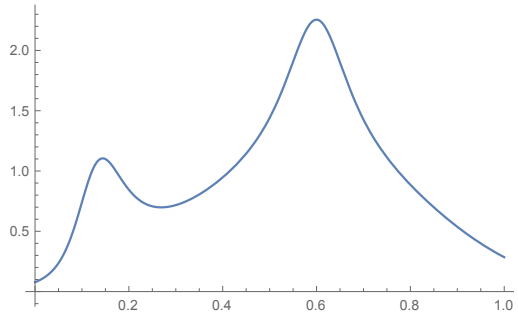
$\beta = 1000$

- Sticking to the exact bound given by Geman–Geman is incredibly slow.
- Geman–Geman's theorem assumes that the starting candidate is random noise.

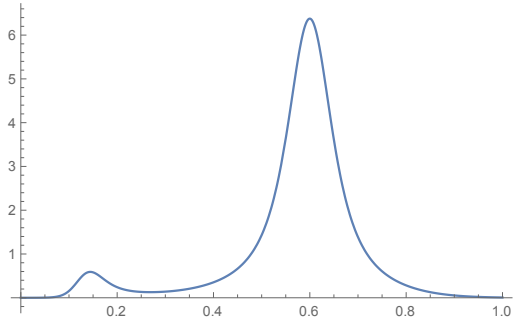
MODIFIED COOLING SCHEDULE



$\beta = 1$



$\beta = 300$



$\beta = 1000$

- Sticking to the exact bound given by Geman–Geman is incredibly slow.
- Geman–Geman's theorem assumes that the starting candidate is random noise.

We introduce parameter τ to control our cooling schedule:

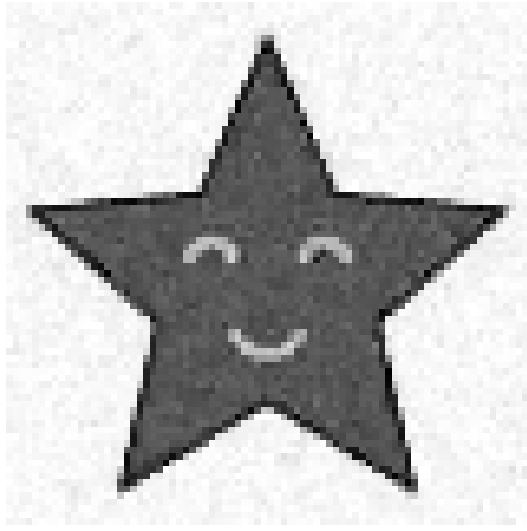
$$\beta_n = \tau \log n$$

GIBBS SAMPLER RESULTS

Original image

Noisy Data

MAP estimate



More general sampler

Metropolis–Hastings:

- Proposal distribution
- Acceptance probability

Special case: Gibbs Sampler

Steps:

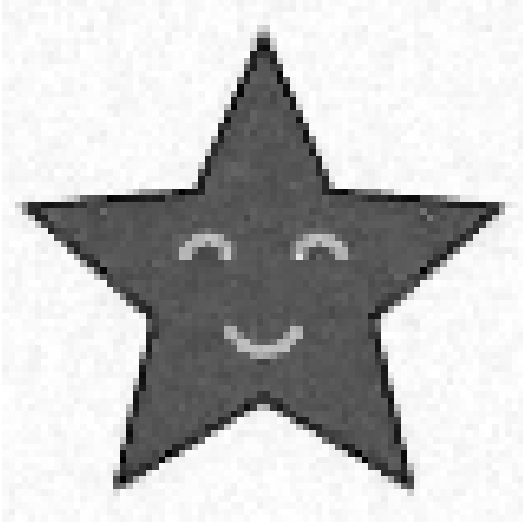
- Randomly choose a pixel s to update.
- Propose image x' by updating $x'_s := x_s \pm 1/k$.
- Compute $\Delta H = H(x') - H(x) = h_{x'}(s) - h_x(s)$.
  - If $\Delta H < 0$, accept the proposed pixel change.
  - Otherwise, accept with probability $\Pi(x')/\Pi(x) = e^{-\beta \Delta H}$.
- Repeat.

RANDOM WALK RESULTS

Original image

Noisy Data

MAP estimate

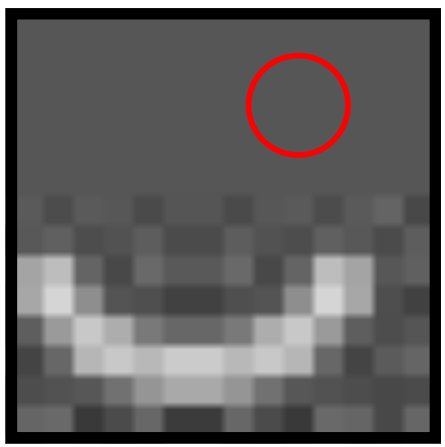


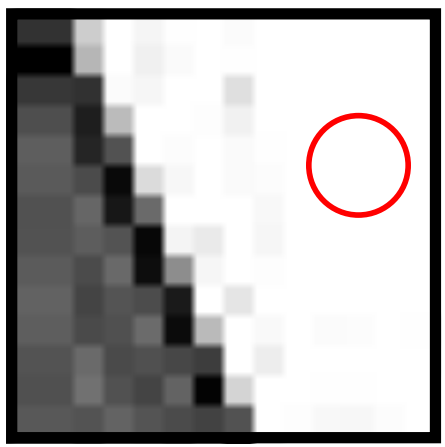
CONSIDERING SPATIAL DEPENDENCE

- Gaussian model isn't perfect—independence
- Spatial correlation

CONSIDERING SPATIAL DEPENDENCE

- Gaussian model isn't perfect—independence
- Spatial correlation
- High consistency around a given pixel in the observed image $\Rightarrow$ probably not much JPEG noise at this point





QUANTIFYING TRUST

Assign a “trust” level to each pixel in the observed image:
$T(s) \in [0, 1]$.

We do so by quantifying the consistency in pixel values around s.

Assign a “trust” level to each pixel in the observed image: $T(s) \in [0, 1]$.

We do so by quantifying the consistency in pixel values around s.

We use:

$$T(s) = 1 - \frac{1}{8(2\kappa - \kappa^2)} \sum_{t \sim s} M(|y_s - y_t|),$$

where

$$M(\delta) = \begin{cases} \delta^2 & \delta < \kappa \\ \kappa^2 + 2\kappa(\delta - \kappa) & \text{otherwise} \end{cases}$$

HOW WE USE TRUST

Main Idea: If we trust a pixel, we should be less willing to update it in a way that increases the image's energy.

Main Idea: If we trust a pixel, we should be less willing to update it in a way that increases the image's energy.

Recall our acceptance probability, $e^{-\beta \Delta H}$.

Instead, we'll use $(1 - T(s))\, e^{-\beta \Delta H}$.

RANDOM WALK WITH TRUST RESULTS

Original image

Noisy Data

MAP estimate

PARAMETERS

Our algorithm contains multiple parameters:

$$\theta, \gamma, \alpha, \tau, \kappa$$

PARAMETERS

Our algorithm contains multiple parameters:

$$\theta, \gamma, \alpha, \tau, \kappa$$

- $h(s) = f(s) + \theta d(s)$

PARAMETERS

Our algorithm contains multiple parameters:

$$\theta, \gamma, \alpha, \tau, \kappa$$

- $h(s) = f(s) + \theta d(s)$
- γ and α tune the neighbor penalty function

PARAMETERS

Our algorithm contains multiple parameters:

$$\theta, \gamma, \alpha, \tau, \kappa$$

- $h(s) = f(s) + \theta d(s)$
- γ and α tune the neighbor penalty function
- τ tunes the cooling schedule

PARAMETERS

Our algorithm contains multiple parameters:

$$\theta, \gamma, \alpha, \tau, \kappa$$

- $h(s) = f(s) + \theta d(s)$
- γ and α tune the neighbor penalty function
- τ tunes the cooling schedule
- κ tunes the trust estimation

The MAP estimate is relatively sensitive to parameter changes.

Optimization for general inputs is a huge problem to tackle.

BACK TO COLOR

Quick and dirty approach:

- Split color images into a grid for R, a grid for G, and grid for B.
- Restore each channel independently.
- Recombine.

A QUICK RECAP

A QUICK RECAP

- Build an energy function $H(x)$ to rate an arbitrary image x with respect to an observed image

A QUICK RECAP

- Build an energy function $H(x)$ to rate an arbitrary image x with respect to an observed image
- Take a probabilistic approach to find the minimizer of H

A QUICK RECAP

- Build an energy function $H(x)$ to rate an arbitrary image x with respect to an observed image
- Take a probabilistic approach to find the minimizer of H
- Construct a distribution based on $H(x) \rightarrow \Pi(x) \propto e^{-H(x)}$

A QUICK RECAP

- Build an energy function $H(x)$ to rate an arbitrary image x with respect to an observed image
- Take a probabilistic approach to find the minimizer of H
- Construct a distribution based on $H(x) \rightarrow \Pi(x) \propto e^{-H(x)}$
- Sample with simulated annealing (with appropriate cooling schedule) $\rightarrow$ the mode of the distribution (and the minimizer of H)

A QUICK RECAP

- Build an energy function $H(x)$ to rate an arbitrary image x with respect to an observed image
- Take a probabilistic approach to find the minimizer of H
- Construct a distribution based on $H(x) \rightarrow \Pi(x) \propto e^{-H(x)}$
- Sample with simulated annealing (with appropriate cooling schedule) $\rightarrow$ the mode of the distribution (and the minimizer of H)
- Empirical adaptations (cooling schedule modified, trust, etc.)

RESULTS

RESTORED STAR

Noisy data

RESTORED STAR

MAP estimate

RESTORED STAR

Original image

ON A PHOTO?

Noisy data

ON A PHOTO?

MAP estimate

ON A PHOTO?

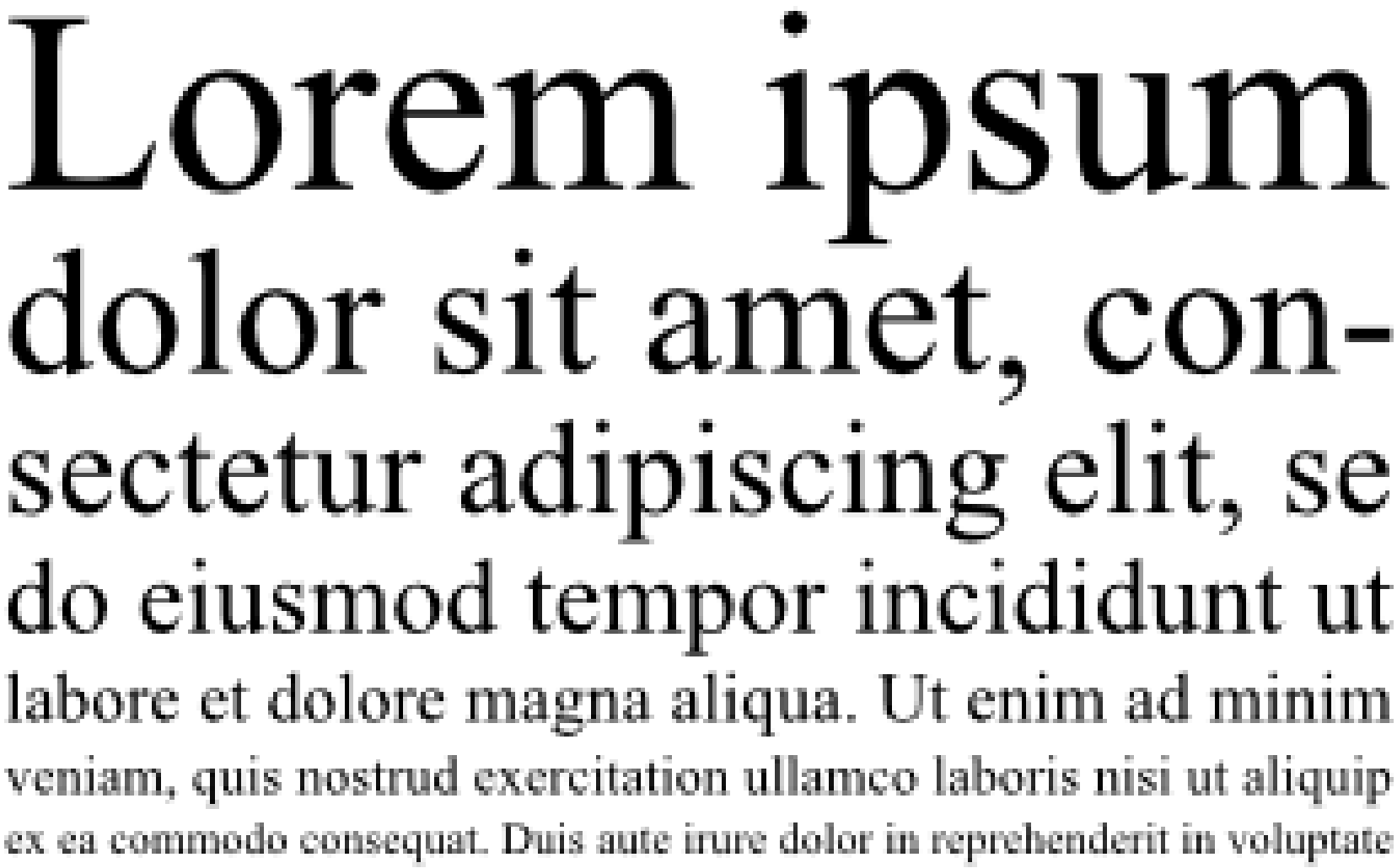
Original image

Noisy data

MAP estimate

Original image

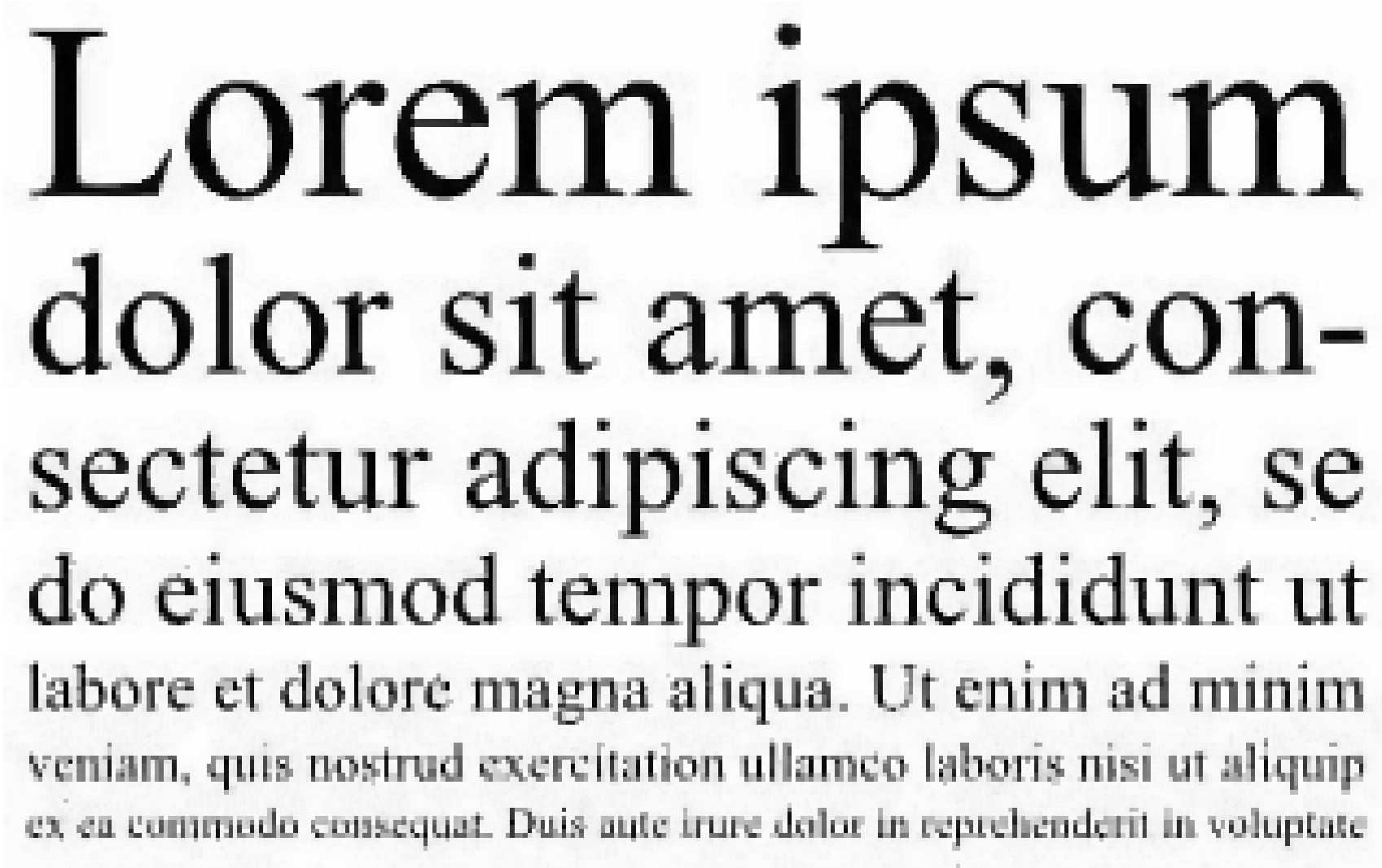
Original image



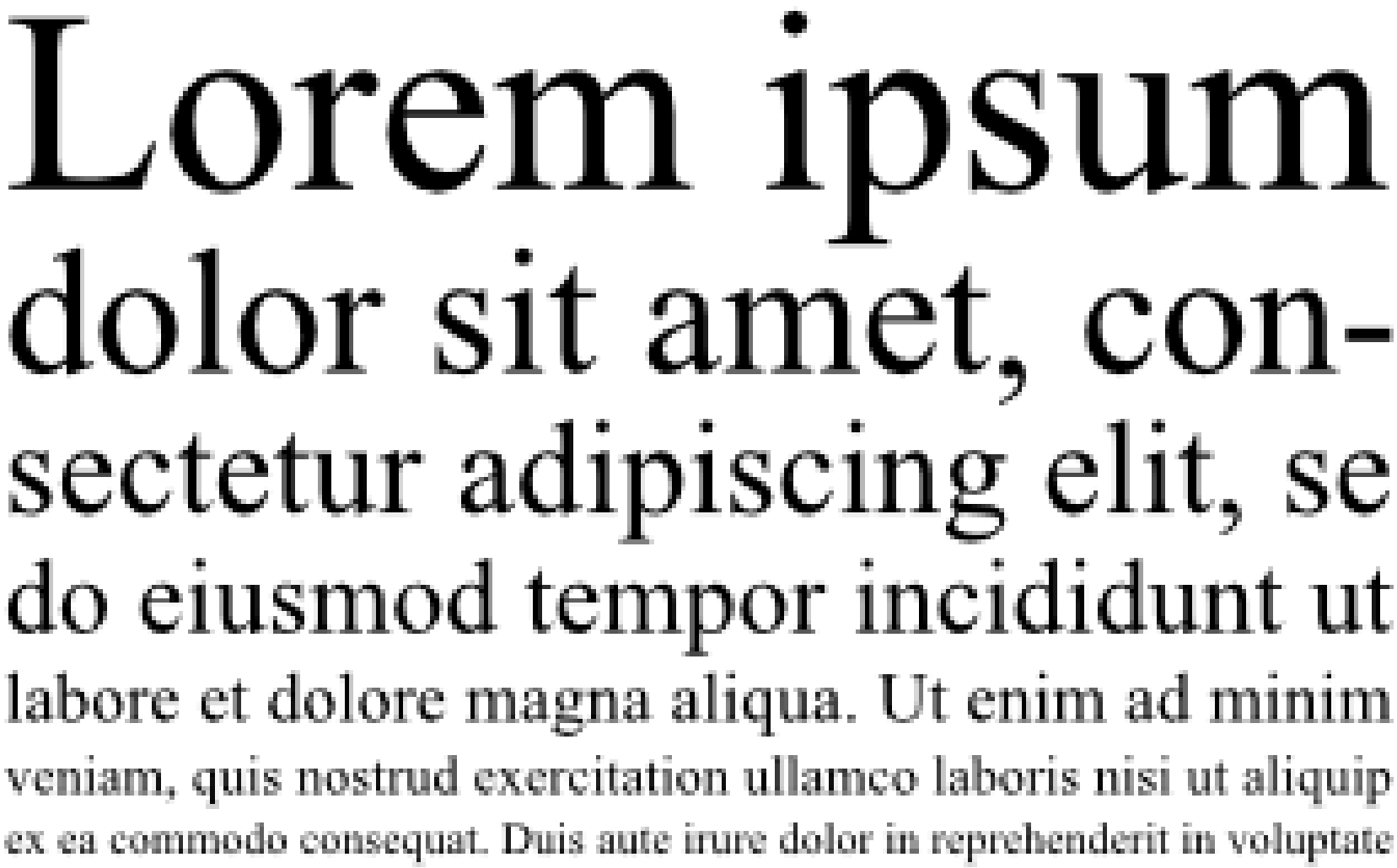
Noisy data


Lorem ipsum
dolor sit amet, con-
sectetur adipiscing elit, se
do eiusmod tempor incididunt ut
labore et dolore magna aliqua. Ut enim ad minim
veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip
ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate

MAP estimate



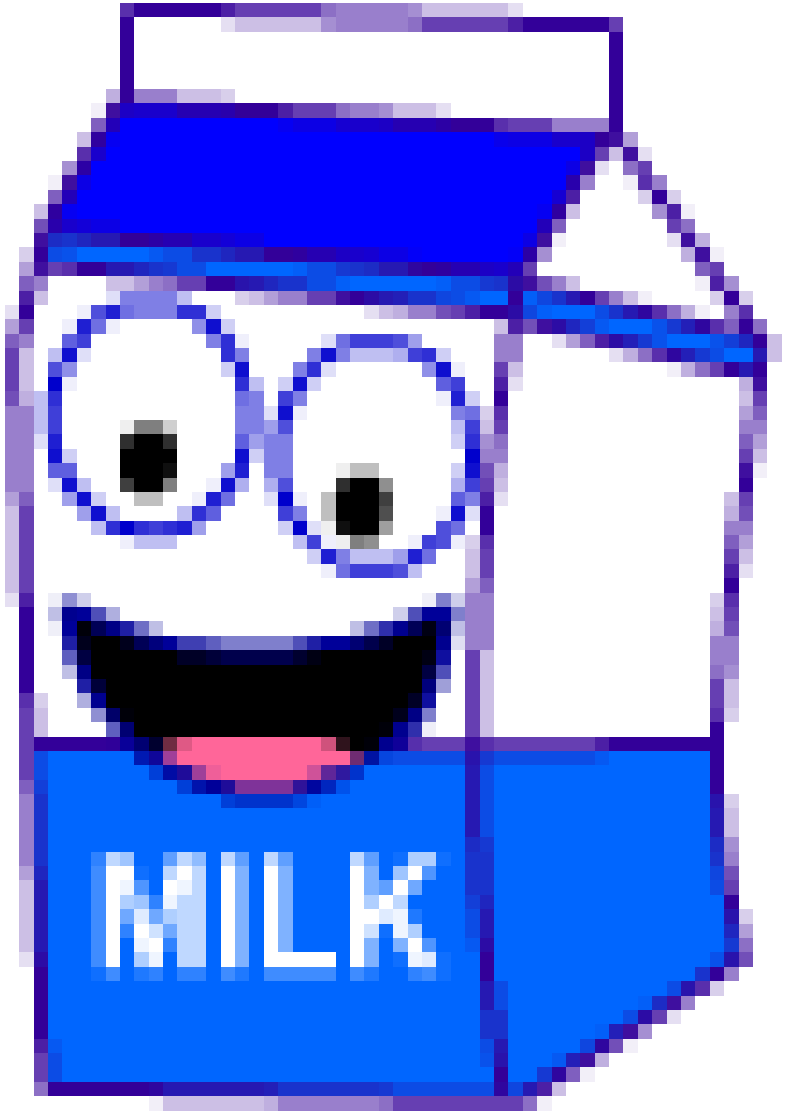
Original image



Noisy data

MAP estimate

Original image



Noisy data

MAP estimate

Original image

Noisy data

MAP estimate

Original image

HOW ABOUT BOB?!

Noisy data

HOW ABOUT BOB?!

MAP estimate

HOW ABOUT BOB?!

Original image

THANK YOU!

Bob Dobrow & the entire Math Department

Those who attended our practice talk

Families and Friends

Markov, Gibbs, Bayes, Geman, other Geman, Hammersley, and Clifford

And **YOU**, our audience!

QUESTIONS?

REFERENCES

”Image Analysis, Random Fields and Markov Chain Monte Carlo Methods” by Gerhard Winkler, Springer, 2nd edition, 2003

”Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images”, by Stuart Geman, Donald Geman, 1984

”Simulated Annealing” by Dimitris Bertsimas, John Tsitsiklism, 1993

”Novel Bayesian deringing method in image interpolation and compression using a SGLI prior” by Cheolkon Jung, Licheng Jiao, 2010

”Improved image decompression for reduced transform coding artifacts” by Thomas P. O’Rourke, Robert L. Stevenson, 2014

”Artifact Reduction in Low Bit Rate DCT-Based Image Compression” by Jiebo Luo, Chang Wen Chen, Kevin J. Parker, and Thomas S. Huang, 1996