Statistical Implications

Dr. Daniel B. Rowe
Professor of Computational Statistics
Department of Mathematical and Statistical Sciences
Marquette University

Outline

Introduction

Univariate Statistics

Bivariate Statistics

Time Series Statistics

Image Statistics

Discussion

Homework

Introduction

When we perform operations on observed data that have a statistical distribution, we change the new observations statistical properties.

When we perform liner operations (linear combination), we change properties such as the mean, variance, and spatio-temporal correlation.

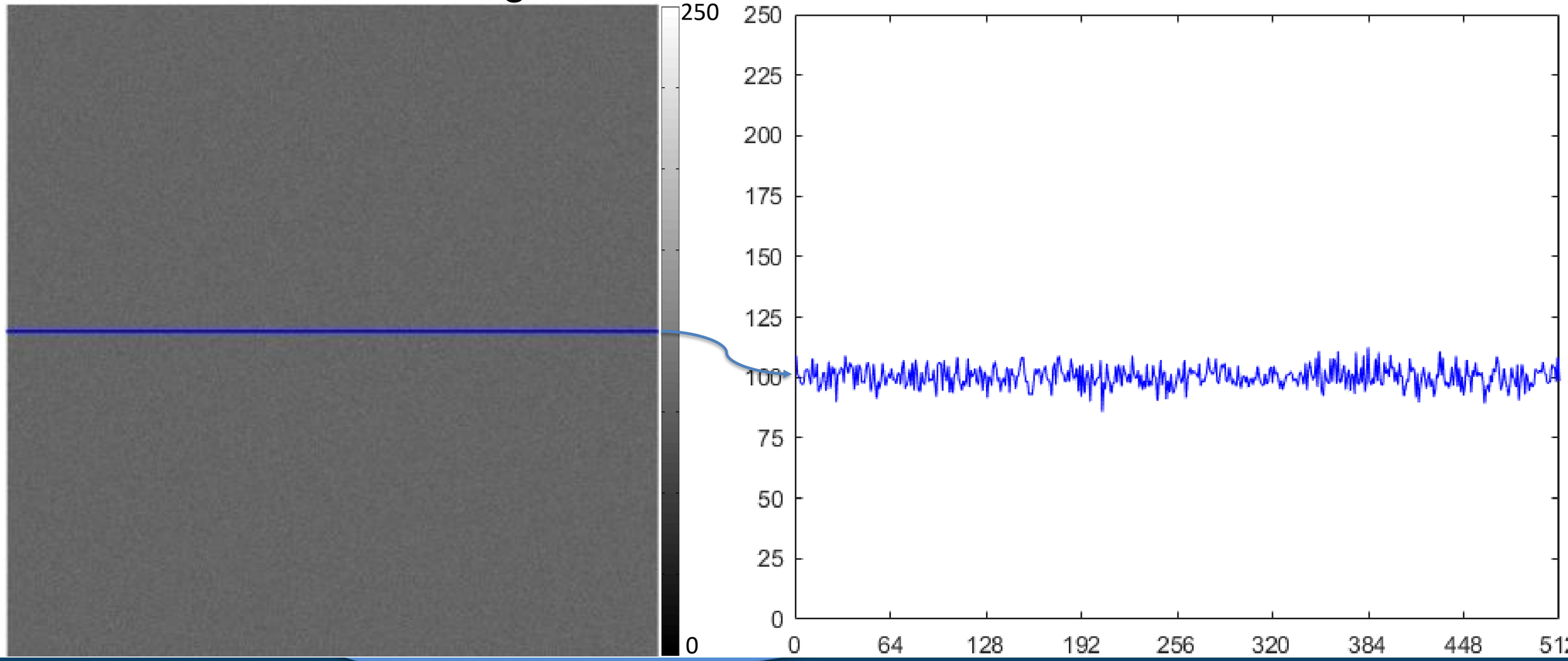
If we perform non-linear operations we also change the distribution.

Let’s examine what happens when we perform linear convolution.

Introduction

$\mu=100, \sigma^2=20$

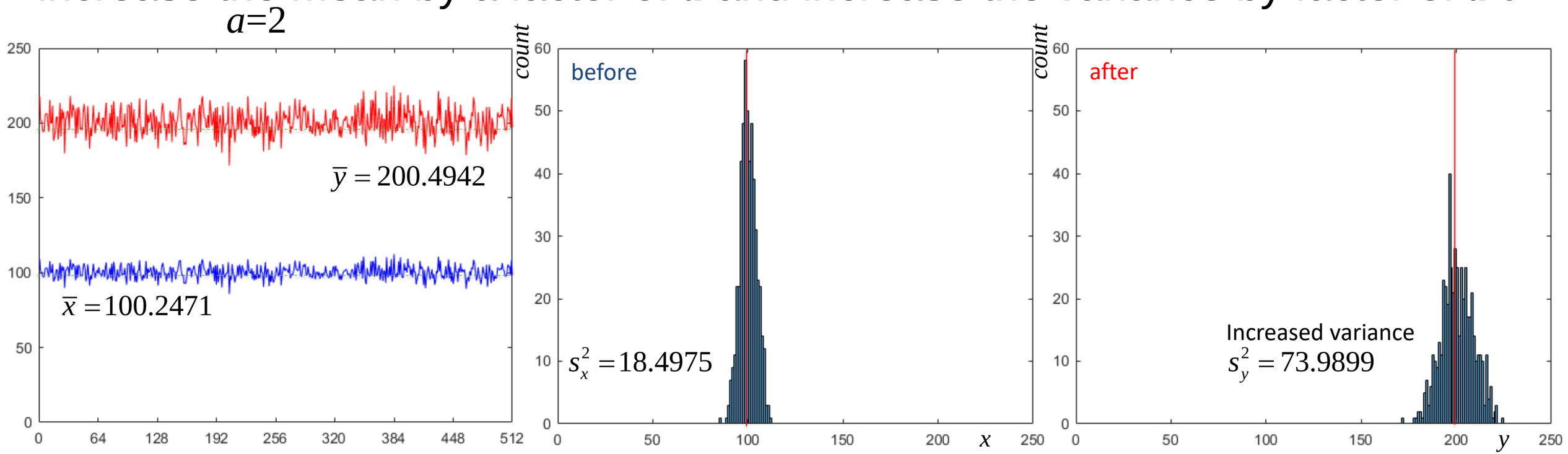
Consider a 512×512 image and let's take the 257th row. A time series.



Introduction

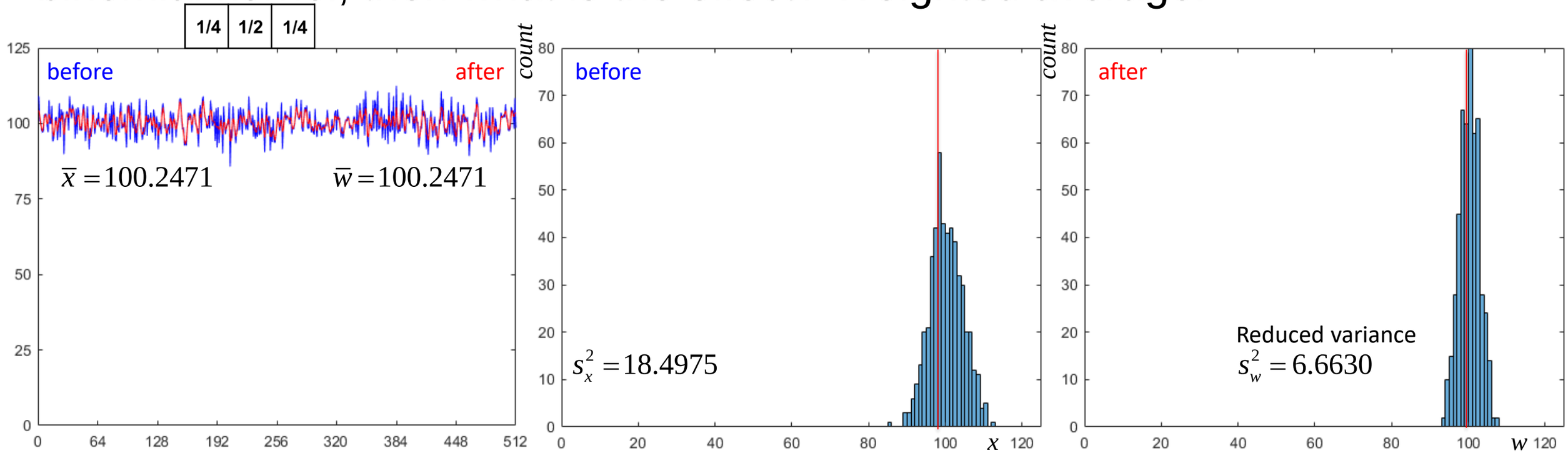
time series

If we have a row of pixels and we multiply all the values by a, then we increase the mean by a factor of a and increase the variance by factor of a^2.



Introduction

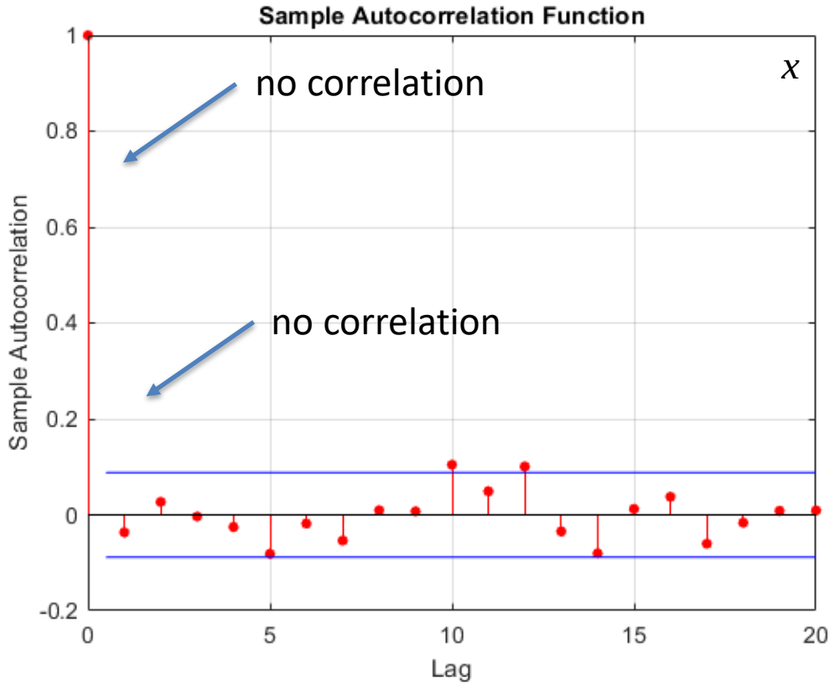
If we were to smooth the time series (row of pixels) with say a 3 point binomial kernel, then what is the effect? Weighted average.

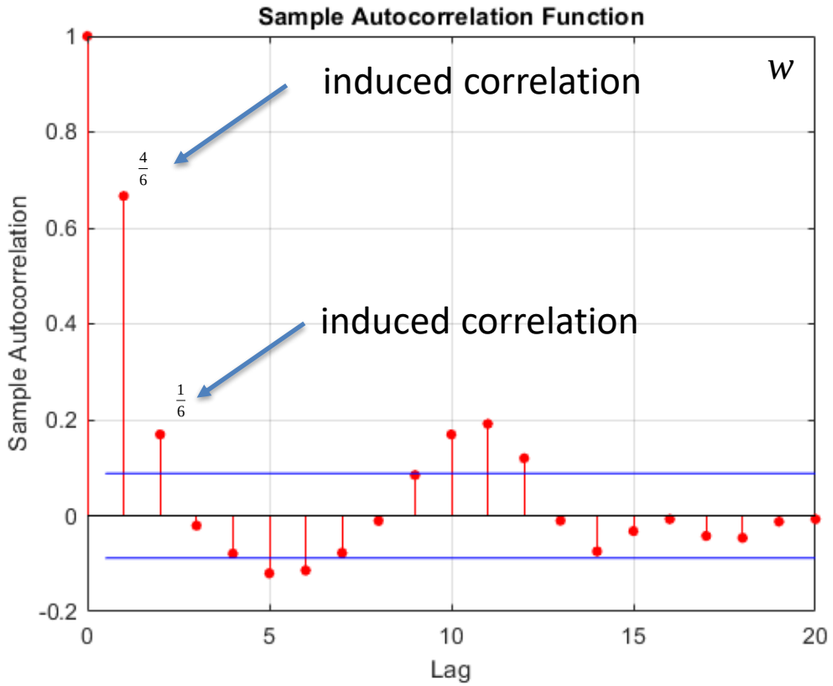


Reduced variance but induced temporal correlation.

Introduction

We can calculate the temporal autocorrelation and see induced correlation.





Make a copy of time series shifted by lag L with wrap around, then calculate The correlation between the two time series.

L	$corx$	$corw$	
0	1.0000	1.0000	
1	-0.0382	**0.6707**	$\frac{4}{6}$
2	0.0344	**0.1757**	$\frac{1}{6}$
3	-0.0049	-0.0162	
4	-0.0210	-0.0768	
5	-0.0791	-0.1179	
6	-0.0147	-0.1125	
7	-0.0577	-0.0785	
8	0.0072	-0.0151	
9	0.0022	0.0800	
10	0.1061	0.1681	
11	0.0519	0.1997	
12	0.1088	0.1384	
13	-0.0220	0.0073	
14	-0.0805	-0.0661	
15	0.0098	-0.0304	
16	0.0428	-0.0082	
17	-0.0609	-0.0446	
18	-0.0187	-0.0516	
19	0.0025	-0.0211	
20	0.0056	-0.0209	

Let's examine how and why does this happen?
This also happens in a 2D image!

Introduction

```
rng('default')
n=512; mu=100; sigma2=20;
limMin=0; limMax=250;

N=sqrt(sigma2)*randn([n,n]);
I=mu+N; x=I(n/2+1,:);
xbar=mean(x), s2x=var(x)

figure;
imagesc(I,[limMin,limMax])
axis image, axis off, colormap(gray)
line([0,n+.5]      ,[n/2+.5,n/2+.5]     ,'Color',[0,0,1],'LineWidth',.01)
line([0,n+.5]      ,[n/2+1+.5,n/2+1+.5],'Color',[0,0,1],'LineWidth',.01)
line([0+1/2,0+1/2],[n/2+.5,n/2+1+.5]  ,'Color',[0,0,1],'LineWidth',.01)
line([n+.5,n+.5]   ,[n/2+.5,n/2+1+.5]  ,'Color',[0,0,1],'LineWidth',.01)
figure;
plot(x,'b')
xlim([0,n]),ylim([limMin,limMax])
set(gca,'xtick',[0:n/8:n]),set(gca,'ytick',[0:limMax/10:limMax])

figure;
histogram(x,25)
xlim([limMin,limMax]), ylim([0,60])
```

Introduction

```
y=2*x;

ybar=mean(y)
y2y=var(y)

figure;
plot(x,'b')
hold on
plot(y,'r')
xlim([0,n]),ylim([limMin,limMax])
set(gca,'xtick',[0:n/8:n])
set(gca,'ytick',[0:50:limMax])

figure;
histogram(y,50)
xlim([limMin,limMax])
Ylim([0,60])
```

```
limMax=125;
kernel=[1/4,1/2,1/4];
ywrap=[x(1,n),x,x(1,1)];
w=zeros(1,n+2);
for i=2:n+1
    w(1,i)=sum(ywrap(1,i-1:i+1).*kernel);
end
w=w(1,2:n+1);
wbar=mean(w),s2w=var(w)

figure;
plot(x,'b')
hold on
plot(w,'r')
xlim([0,n]),ylim([limMin,limMax])
set(gca,'xtick',[0:n/8:n])
set(gca,'ytick',[0:25:limMax])

figure;
histogram(w,20)
xlim([limMin,limMax])
```

```
figure;
autocorr(x)
figure;
autocorr(w)
```

```
% autocorrelation
numlags=20;
lagcorx=zeros(numlags,1);
lagcorw=zeros(numlags,1);
for t=1:numlags
    xx=circshift(x,t);
    temp=corrcoef(x,xx);
    lagcorx(t,1)=temp(1,2);
    ww=circshift(w,t);
    temp=corrcoef(w,ww);
    lagcorw(t,1)=temp(1,2);
end
[(1:numlags)',lagcorx,lagcorw]
```

Univariate Statistics

Let’s discuss the statistical methodology of why and how this happens.

We will start small, only discuss enough to understand how and why.

In doing this, some statistics math will be utilized to in addition to pictures to help understand.

The mathematics is not important, but understanding what it telling us is.

Univariate Statistics

In statistics if we observe a random variable x like a pixel value that has a mean (expectation) of μ and variance of σ^2, we write this as:

$E(x)=\mu$ and $var(x)=\sigma^2$.

If we multiply our observed random variable x by a constant a, then

$E(ax)=a\mu$ and $var(ax)=a^2\sigma^2$.

Just think of this as a rule.

i.e. $x \sim N(\mu,\sigma^2)$, then $y=ax \sim N(a\mu,a^2\sigma^2)$

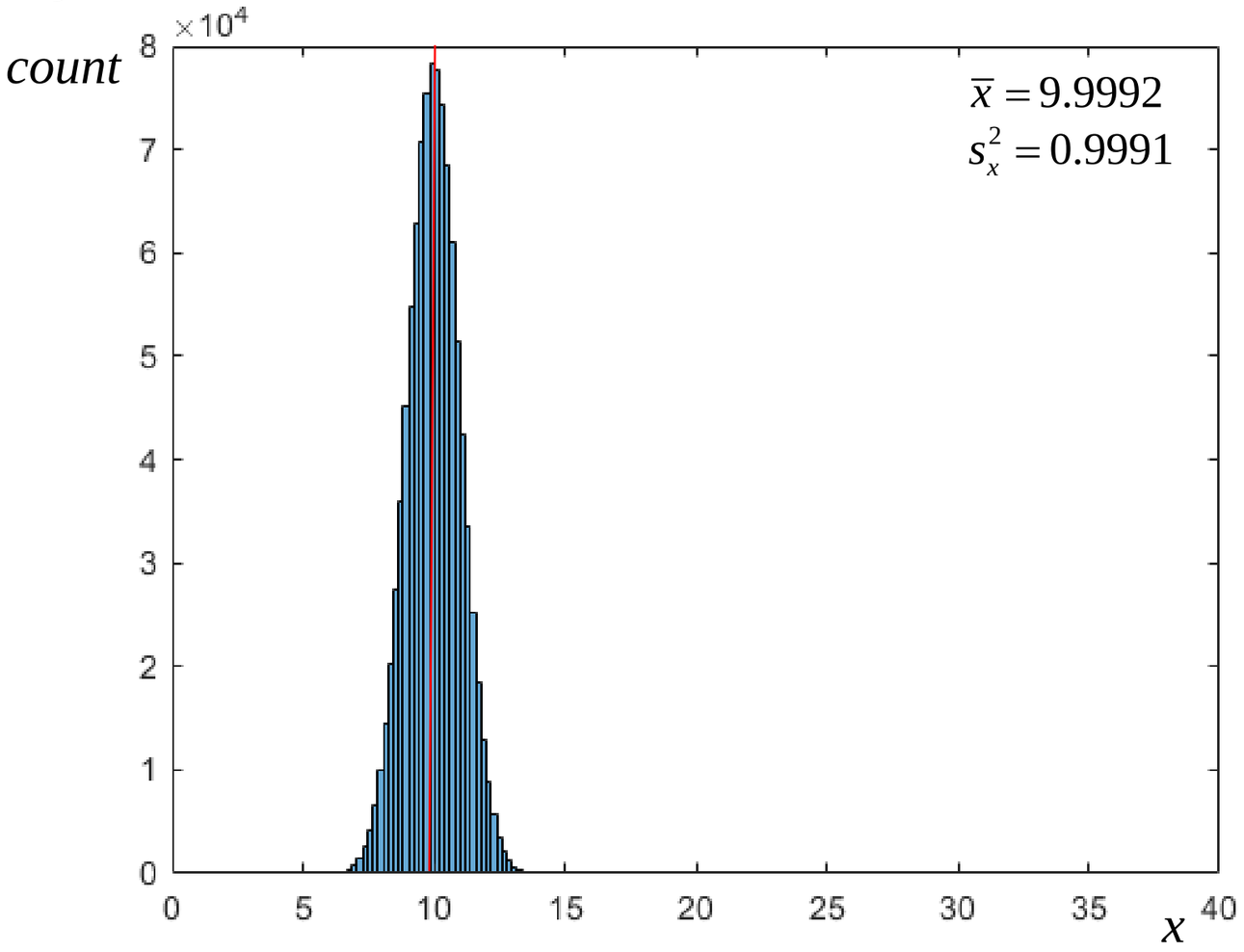
Univariate Statistics

$E(ax)=a\mu$ and $var(ax)=a^2\sigma^2$.

Example: Observe same pixel many times.

$x \sim N(10,1)$

```
n=10^6;
mu=10; sigma2=1;

x=sqrt(sigma2)*randn([n,1])+mu;
xbar=mean(x)
s2x=var(x)

figure;
histogram(x,50)
xlim([0,40])
```



Univariate Statistics

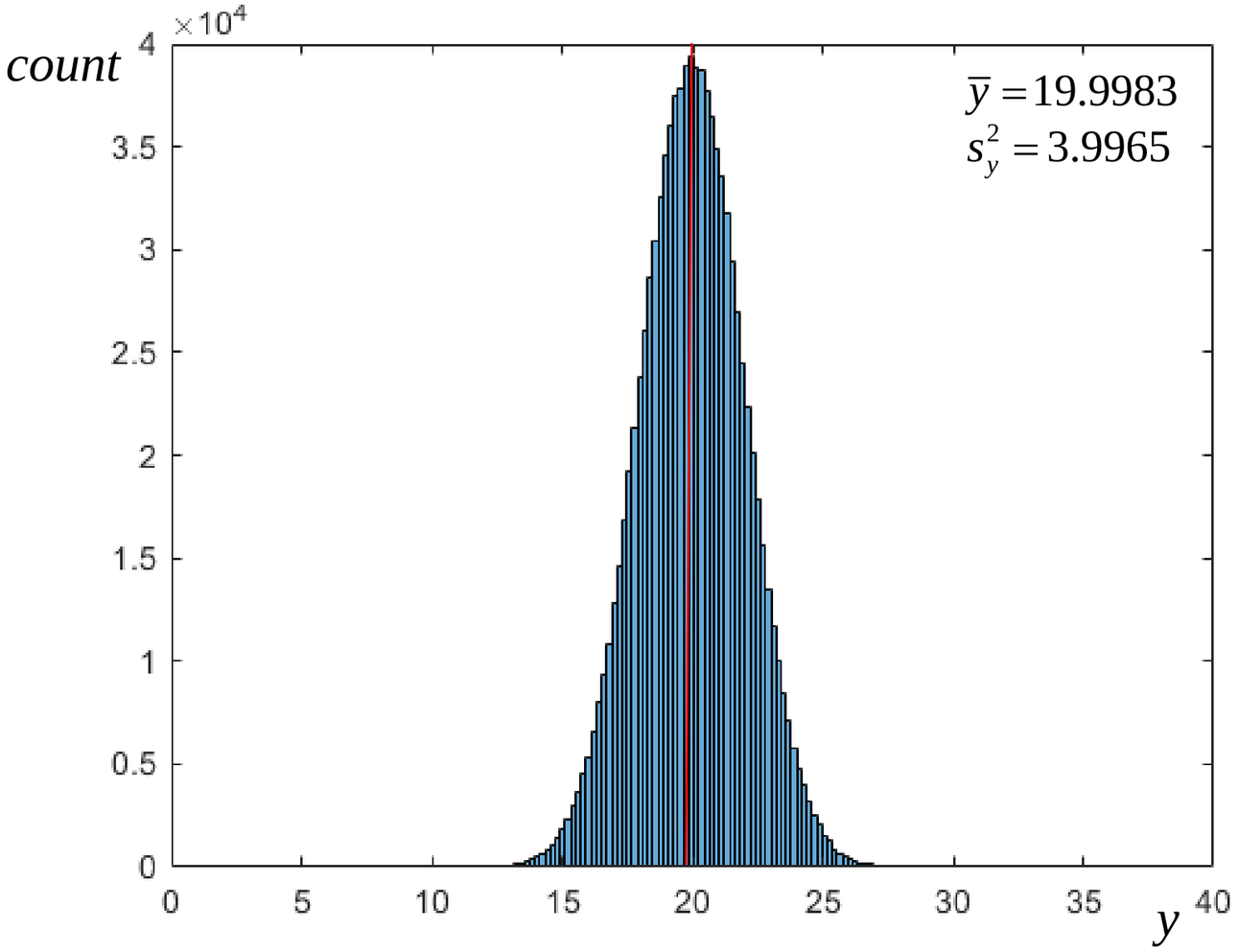
$E(ax)=a\mu$ and $var(ax)=a^2\sigma^2$.

Example: Observe same pixel many times. Multiply by $a=2$.

$y=2x \sim N(20,4)$

Changed mean and variance.

```
y=2*x;
ybar=mean(y)
s2y=var(y)

figure;
histogram(y,100)
xlim([0,40])
```



Univariate Statistics

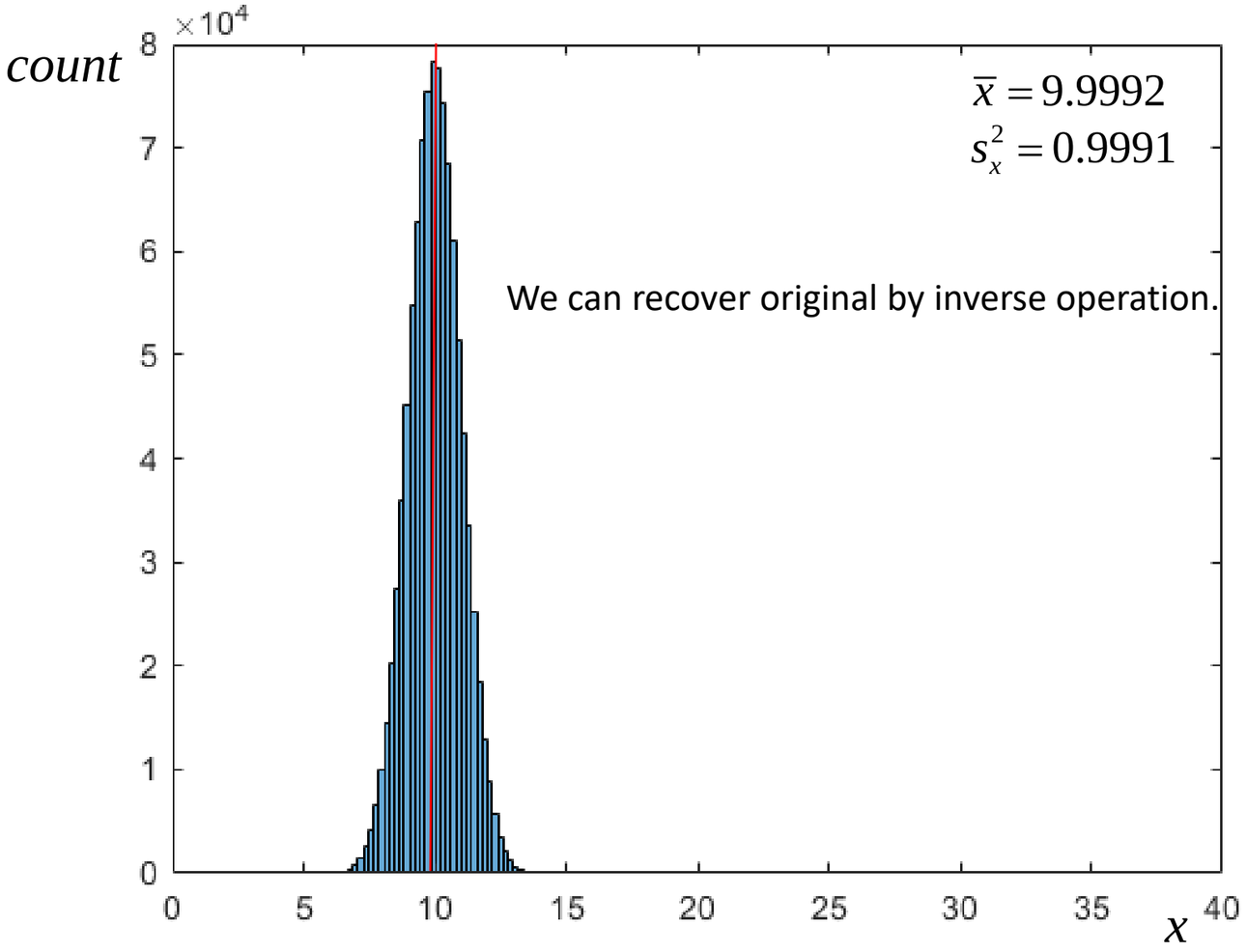
$E(ax)=a\mu$ and $var(ax)=a^2\sigma^2$.

Example: Observe same pixel many times. Divide by $a=2$.

$w=y/2 \sim N(10,1)$

Recovered mean and variance.

```
w=y/2;
wbar=mean(w)
s2y=var(w)

figure;
histogram(w,50)
xlim([0,40])
```



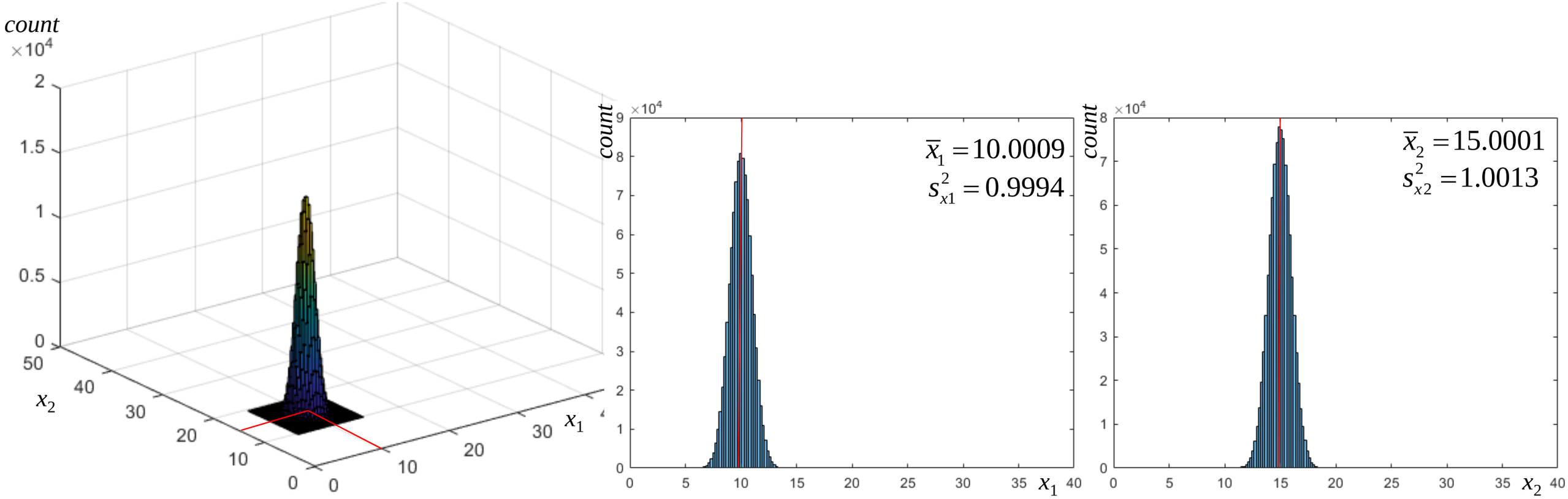
Bivariate Statistics

$E(ax)=a\mu$ and $var(ax)=a^2\sigma^2$.

Example: Observe same two pixels many times.

$x_1 \sim N(10,1)$

$x_2 \sim N(15,1)$



Bivariate Statistics

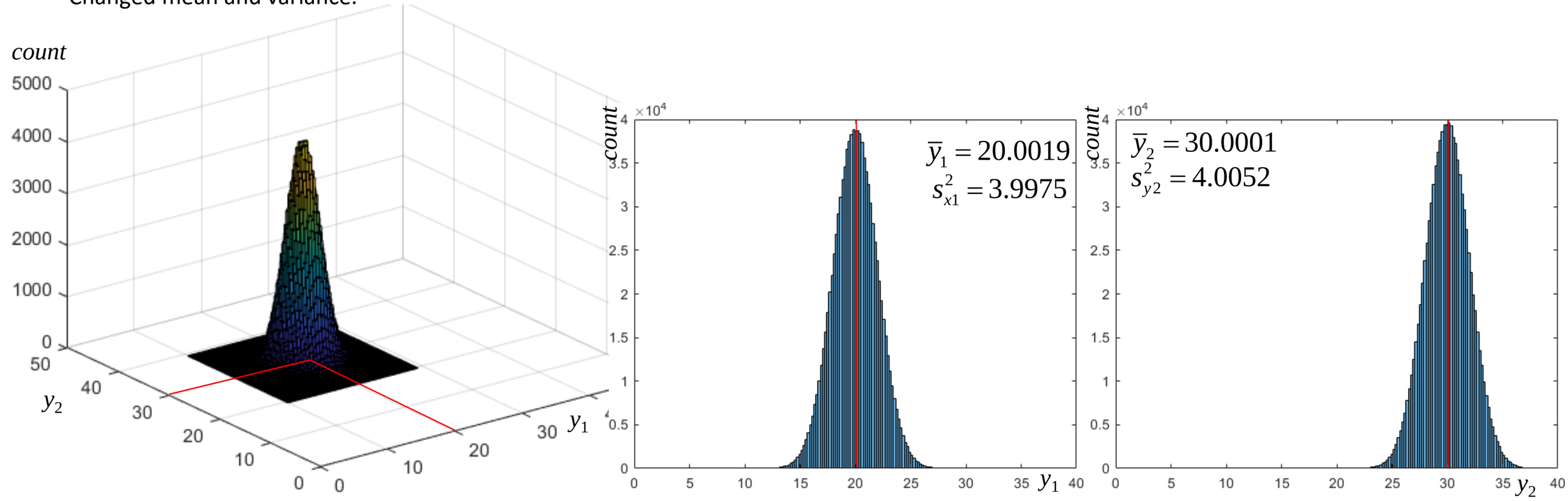
$E(ax)=a\mu$ and $var(ax)=a^2\sigma^2$.

Example: Observe same two pixels many times. Multiply by $a=2$.

$y_1=2x_1 \sim N(20,4)$

$y_2=2x_2 \sim N(30,4)$

Changed mean and variance.



Bivariate Statistics

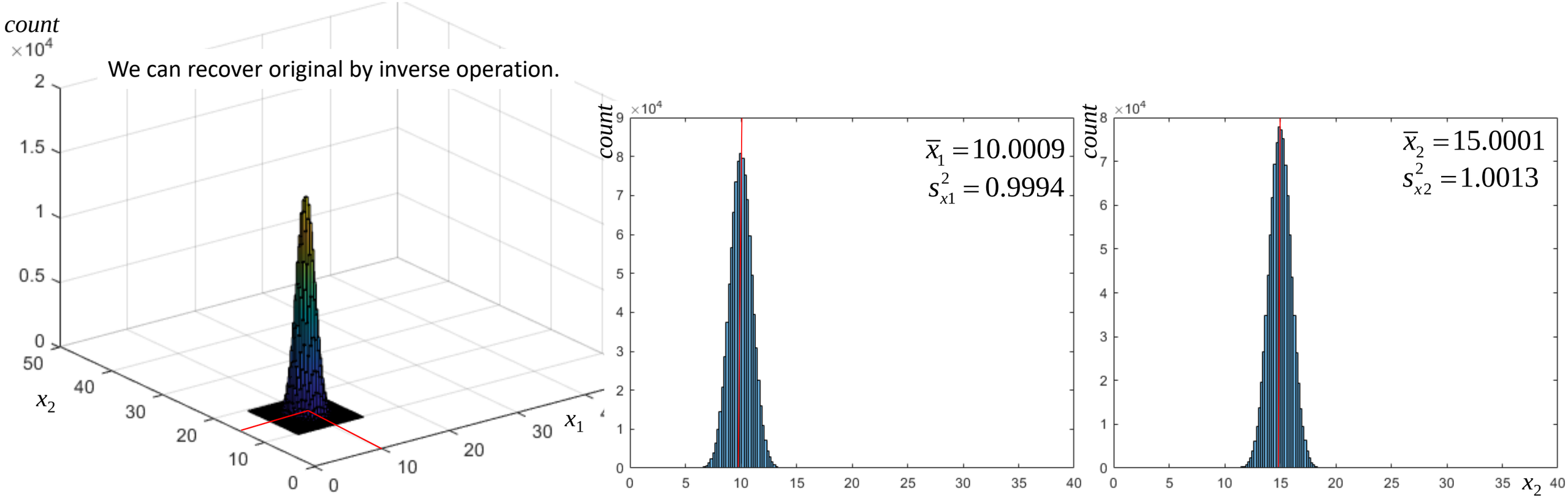
$E(ax)=a\mu$ and $var(ax)=a^2\sigma^2$.

Example: Observe same two pixels many times. Divide by $a=2$.

$x_1=y_1/2\sim N(10,1)$

$x_2=y_2/2\sim N(15,1)$

We can recover original by inverse operation.



Bivariate Statistics

```
rng('default')
n=10^6;

mu1=10;, sigma21=1;
mu2=15;, sigma22=1;

x1=sqrt(sigma21)*randn([1,n])+mu1;
x2=sqrt(sigma22)*randn([1,n])+mu2;

x1bar=mean(x1),s2x1=var(x1)
x2bar=mean(x2),s2x2=var(x2)

figure;
histogram(x1,50),xlim([0,40])
figure;
histogram(x2,50),xlim([0,40])

figure;
hist3([x1',x2'],[30,30],'CDataMode','auto',...
    'FaceColor','interp','EdgeColor',[0,0,0])
xlim([0,50]),ylim([0,50]), view(-37,30)
```

```
y=[2,0;0,2]*[x1;x2];
y1=y(1,:); y2=y(2,:);

y1bar=mean(y1),s2y1=var(y1)
y2bar=mean(y2),s2y2=var(y2)

figure;
histogram(y1,100)
xlim([0,40])

figure;
histogram(y2,100)
xlim([0,40])

figure;
hist3([y1',y2'],[60,60],'CDataMode','auto',...
    'FaceColor','interp','EdgeColor',[0,0,0])
xlim([0,50]), ylim([0,50]), view(-37,30)
```

Bivariate Statistics

Instead of treating pixels x_1 and x_2 separately to produce y_1 and y_2, we can treat them together.

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$ to do the same as $y_1=2x_1$ and $y_2=2x_2$,

$$\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}^{-1} = \begin{bmatrix} ½ & 0 \\ 0 & ½ \end{bmatrix}$$

and perform the inverse operation by

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/2 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

if we know the forward process.

Bivariate Statistics

In addition to multiplying pixels x_1 and x_2 by 2, we can include an off diagonal to produce y_1 and y_2, and continue to treat them together.

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$ to do the same as $w_1=2x_1+x_2$ and $w_2=x_1+2x_2$,

and perform the inverse operation by

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2/3 & -1/3 \\ -1/3 & 2/3 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} 2/3 & -1/3 \\ -1/3 & 2/3 \end{bmatrix} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

if we know the forward process.

Bivariate Statistics

In statistics if we observe a bivariate random variable $\underset{2\times1}{x}$ that has a bivariate mean of $\underset{2\times1}{\mu}$ and covariance of $\underset{2\times2}{\Sigma}$, we write this as:

$\underset{2\times1}{E(x)}=\underset{2\times1}{\mu}$ and $\underset{2\times1}{cov(x)}=\underset{2\times2}{\Sigma}$.

If we multiply our observed bivariate random variable $\underset{2\times1}{x}$ by a matrix $\underset{2\times2}{A}$, then

$\underset{2\times2\,2\times1}{E(Ax)}=\underset{2\times2\,2\times1}{A\mu}$ and $\underset{2\times2\,2\times1}{cov(Ax)}=\underset{2\times2\,2\times2\,2\times2}{A\Sigma A'}$.

Just think of this as a rule.

i.e. $\underset{2\times1}{x}\sim N(\underset{2\times1}{\mu},\underset{2\times2}{\Sigma})$, then $\underset{2\times1}{w}=\underset{2\times2\,2\times1}{Ax}\sim N(\underset{2\times2\,2\times1}{A\mu},\underset{2\times2\,2\times2\,2\times2}{A\Sigma A'})$

Bivariate Statistics

$E(Ax)=A\mu$ and $cov(Ax)= A\Sigma A'$

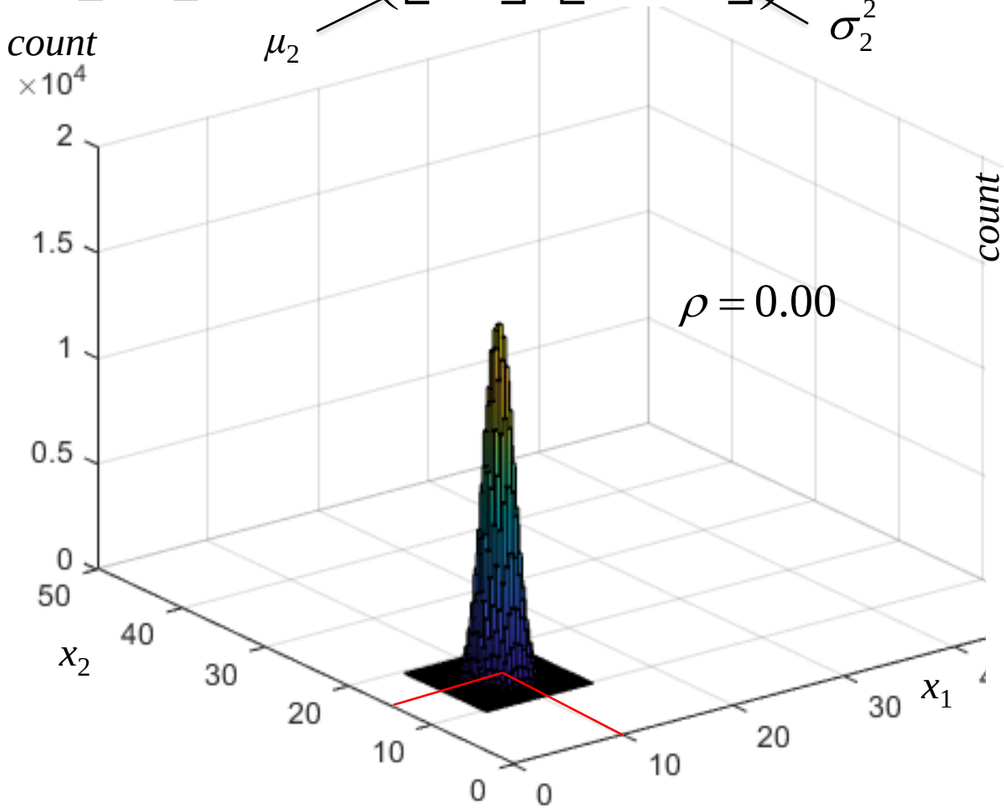
$x_1 \sim N(10,1)$

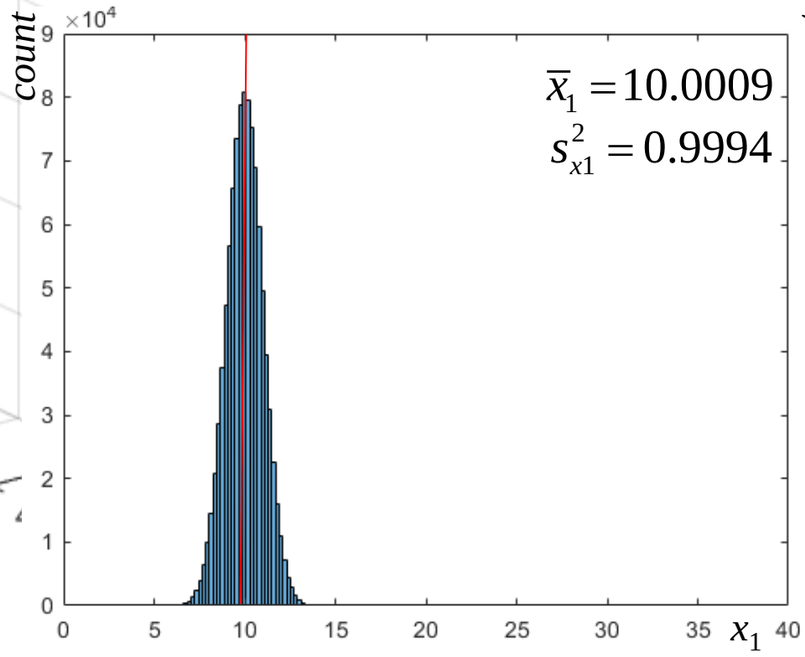
$x_2 \sim N(15,1)$

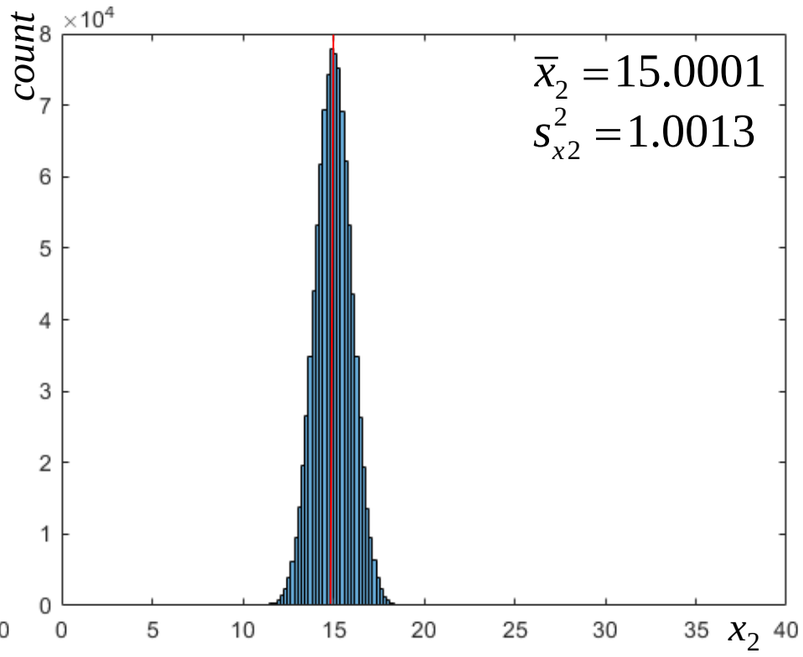
Example:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \sim N\left(\begin{bmatrix} 10 \\ 15 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$$

μ_1, σ_1^2, σ_{12}, μ_2, σ_2^2







Bivariate Statistics

$E(Ax)=A\mu$ and $cov(Ax)= A\Sigma A'$

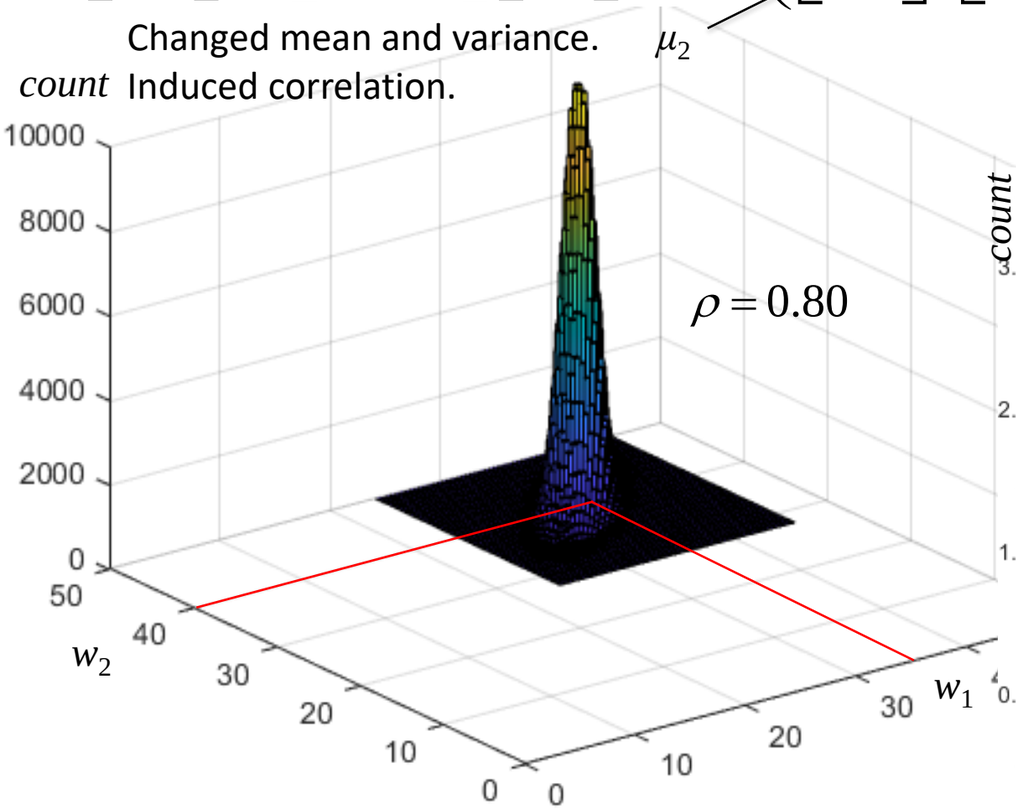
Example:

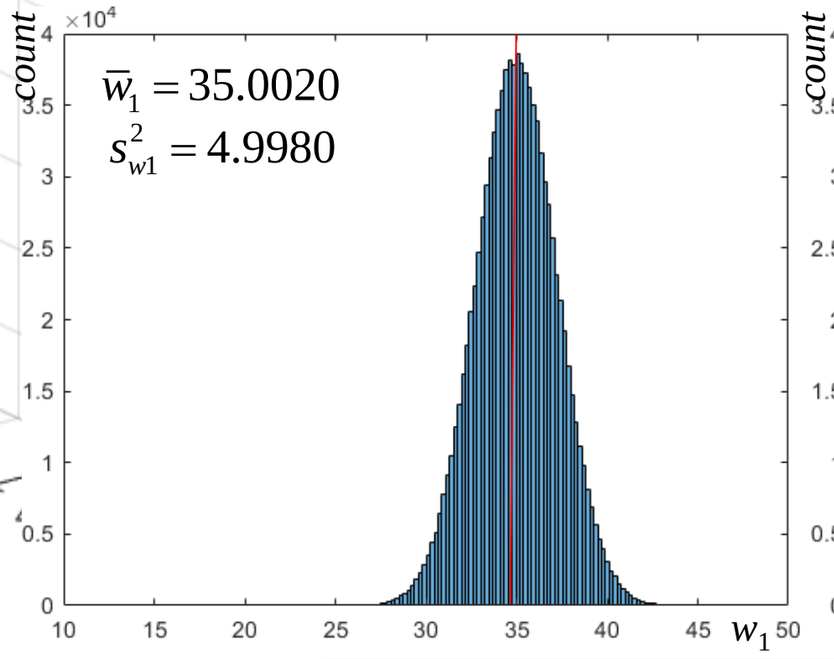
$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \sim N\left(\begin{bmatrix} 35 \\ 40 \end{bmatrix}, \begin{bmatrix} 5 & 4 \\ 4 & 5 \end{bmatrix} \right)$$

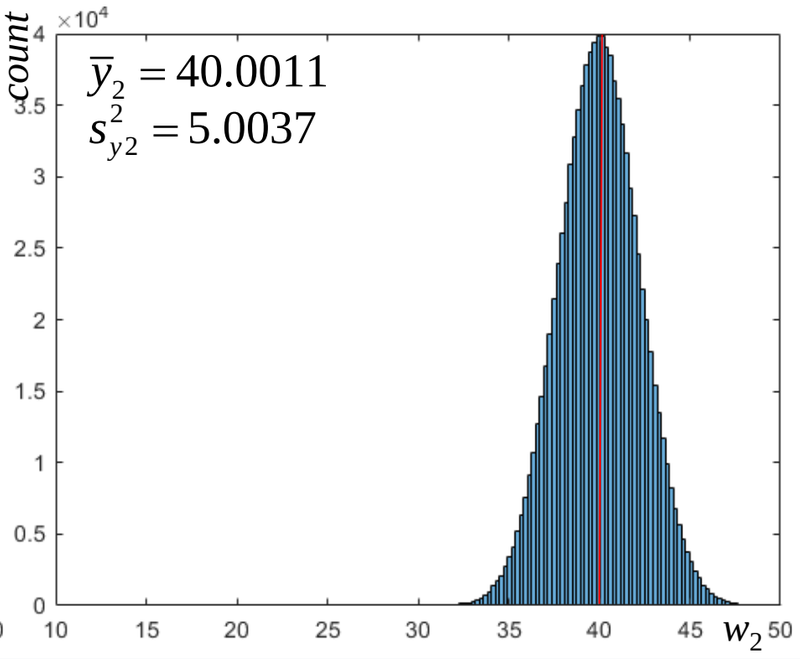
μ_1, σ_1^2, σ_{12}, μ_2, σ_2^2

Changed mean and variance.

Induced correlation.







Bivariate Statistics

$E(Ax)=A\mu$ and $cov(Ax)= A\Sigma A'$

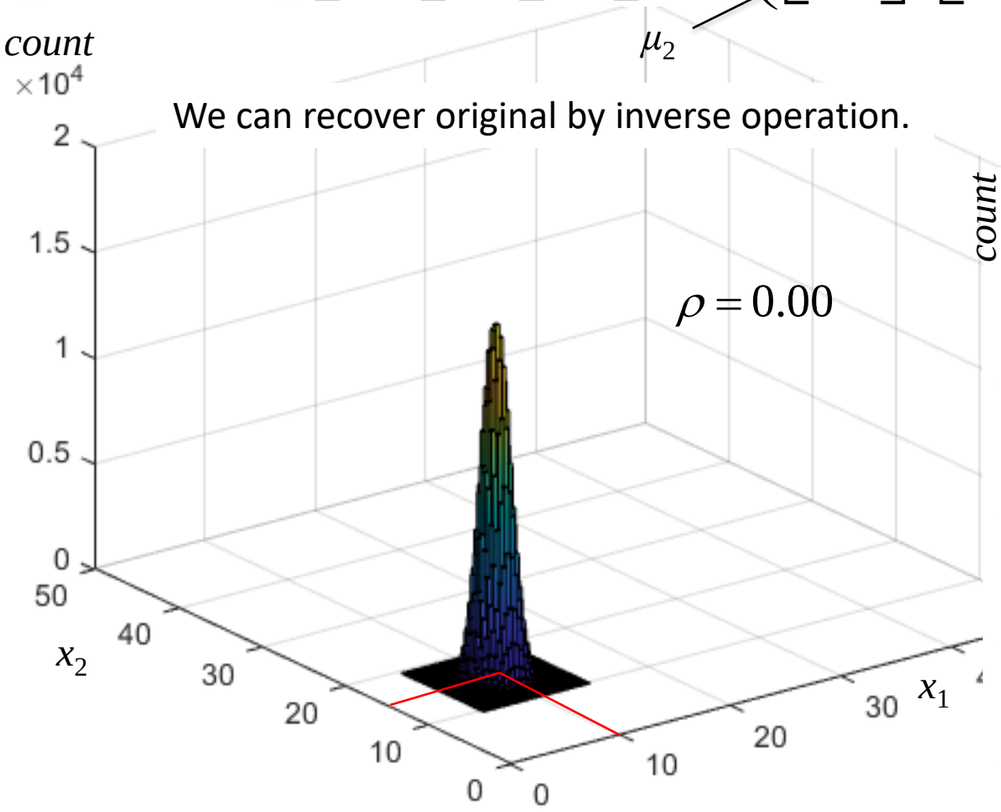
Example:

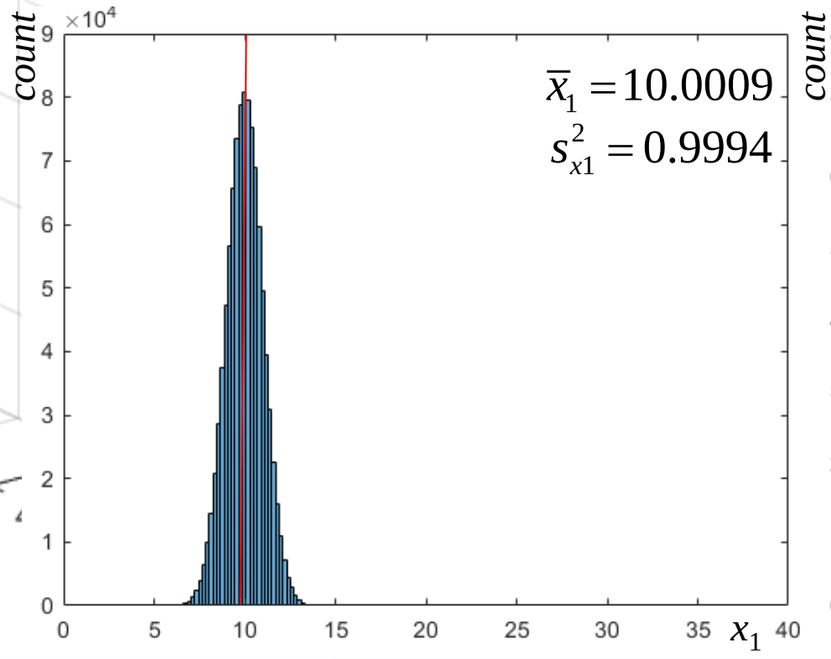
$$\begin{bmatrix} 18/10 & -9/4 \\ -9/4 & 18/10 \end{bmatrix}\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \sim N\left(\begin{bmatrix} 10 \\ 15 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$$

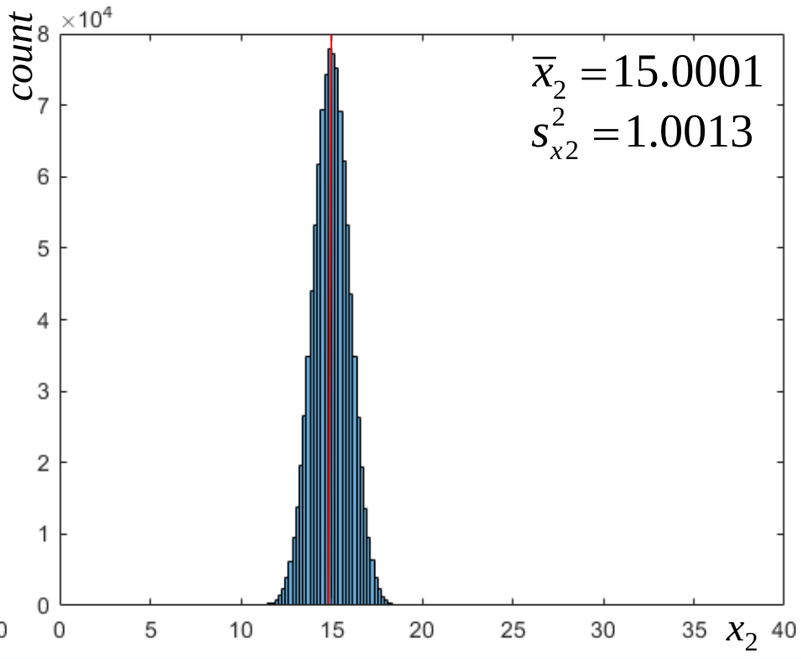
μ_1, σ_1^2, σ_{12}, μ_2, σ_2^2

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}^{-1} = \begin{bmatrix} 18/10 & -9/4 \\ -9/4 & 18/10 \end{bmatrix}$$

We can recover original by inverse operation.







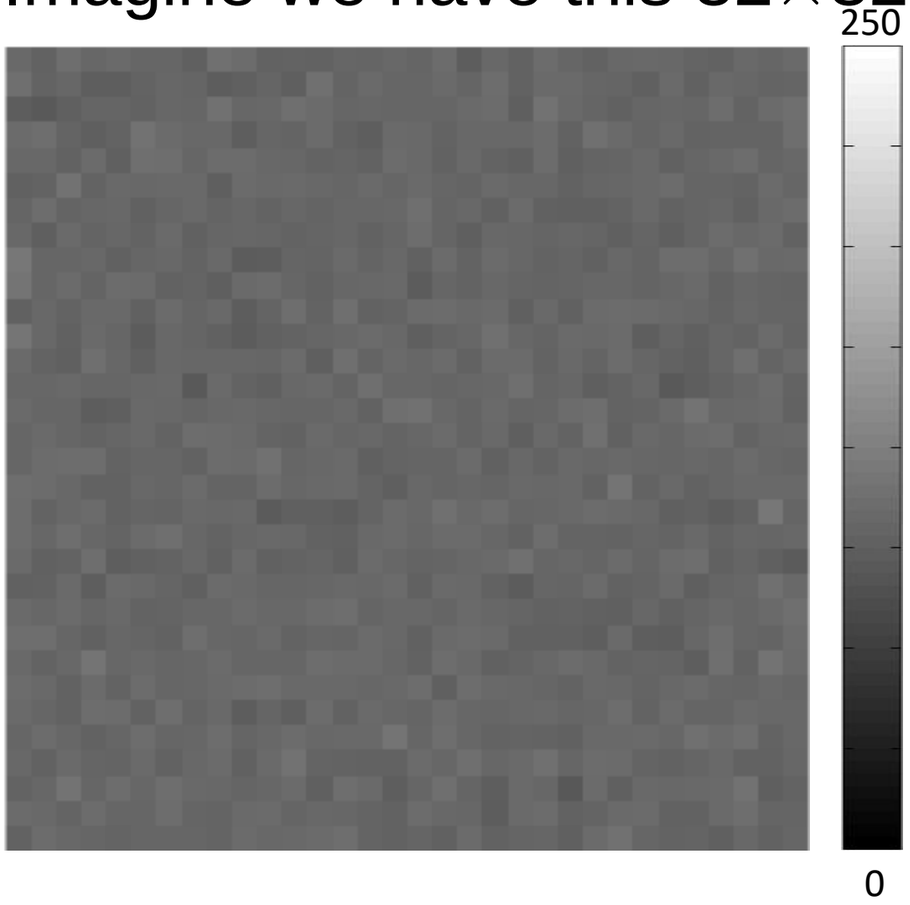
Bivariate Statistics

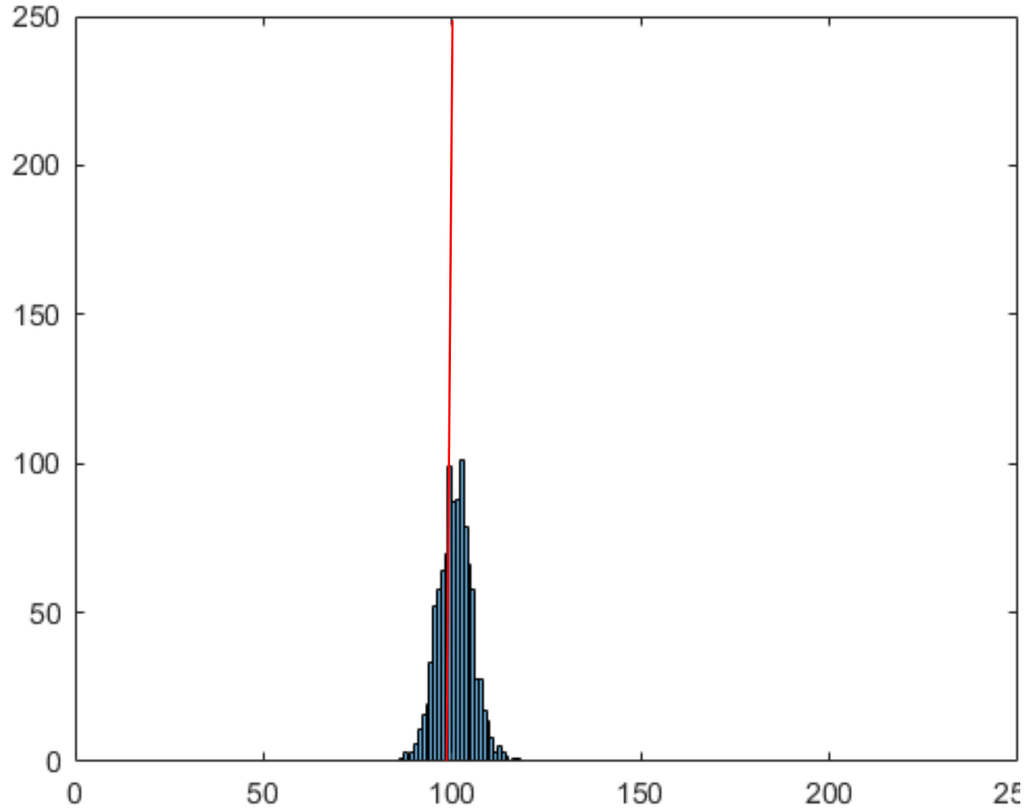
```
A=[2,1;1,2];

w=A*[x1;x2];
w1=w(1,:); w2=w(2,:);

w1bar=mean(w1),s2w1=var(w1)
w2bar=mean(w2),s2w2=var(w2)

figure;
histogram(w1,100)
xlim([10,50])

figure;
histogram(w2,100)
xlim([10,50])

figure;
hist3([w1',w2'],[50,50],'CDataMode',...
    'auto','FaceColor','interp','EdgeColor',[0,0,0])
xlim([0,50]), ylim([0,50]), view(-37,30)
```

Time Series Statistics

$E(x_{ji})=100$ and $var(x_{ji})=20$.

Imagine we have this 32×32 noisy image.





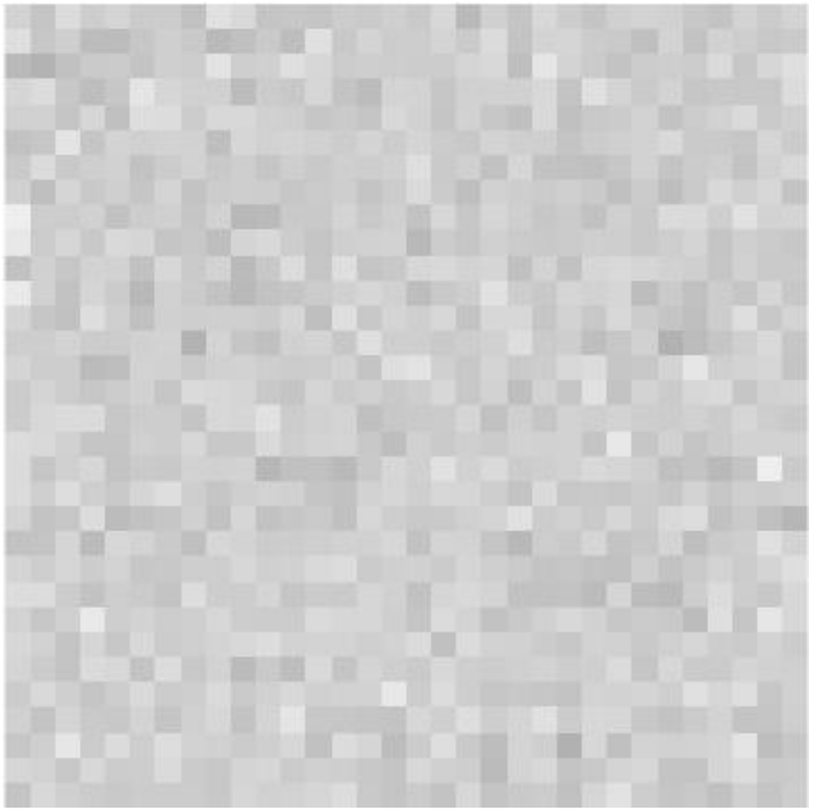
$\bar{I} = 100.8518$

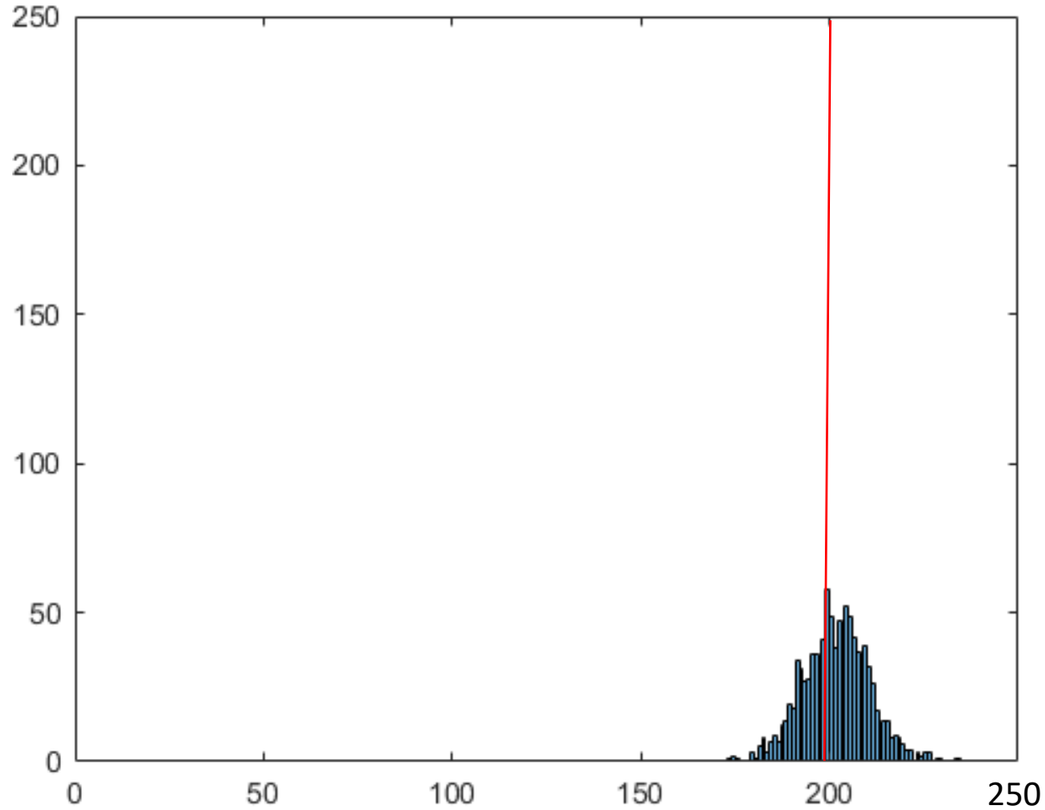
$s_I^2 = 19.8321$

Time Series Statistics

$E(2x_{ji})=200$ and $var(2x_{ji})=80$.

If we multiply every pixel value in the noisy image by 2,





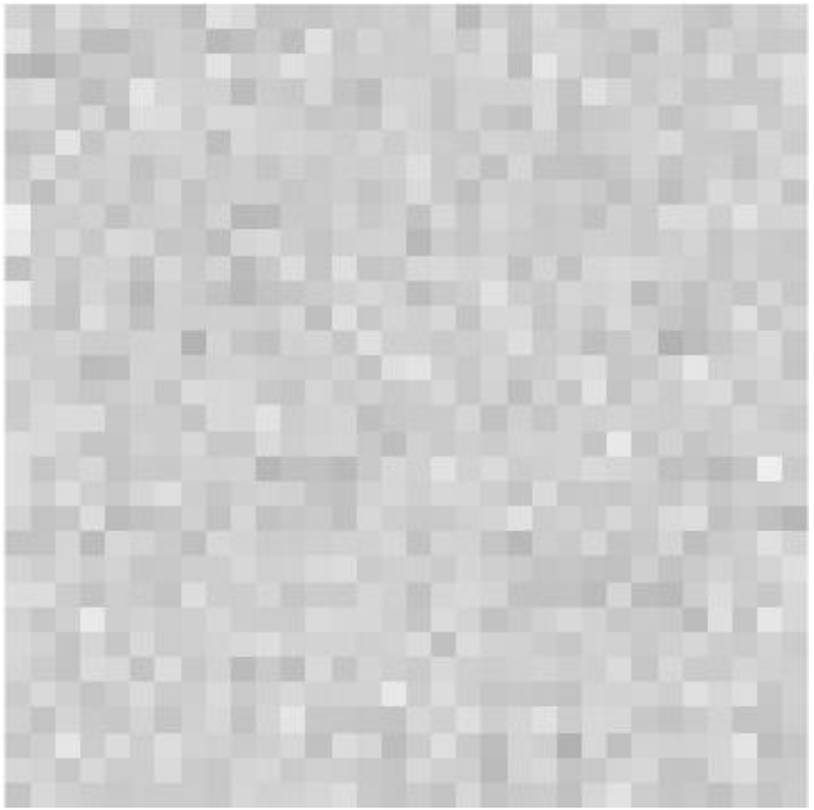
$\bar{I} = 201.7037$
$s_I^2 = 79.3283$

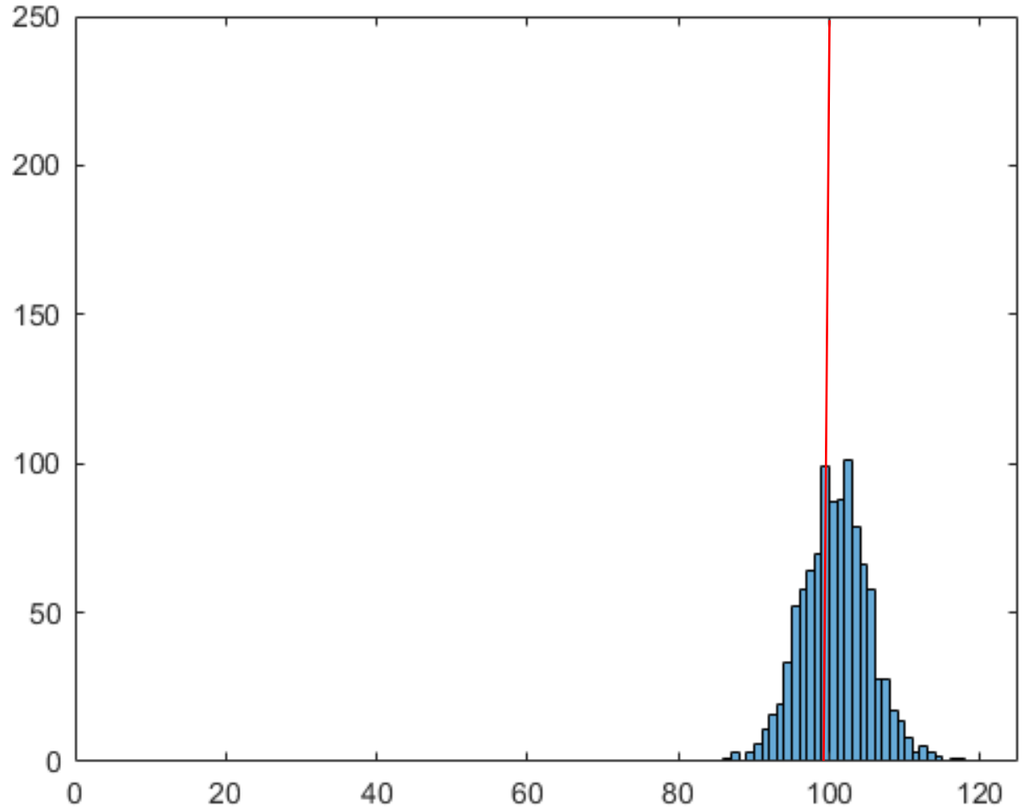
then the average value in the image doubles, and the noise variance quadruples!

Time Series Statistics

$E(x_{ji})=100$ and $var(x_{ji})=20$.

Imagine we have the same 32×32 noisy image.





$\bar{I} = 100.8518$

$s_I^2 = 19.8321$

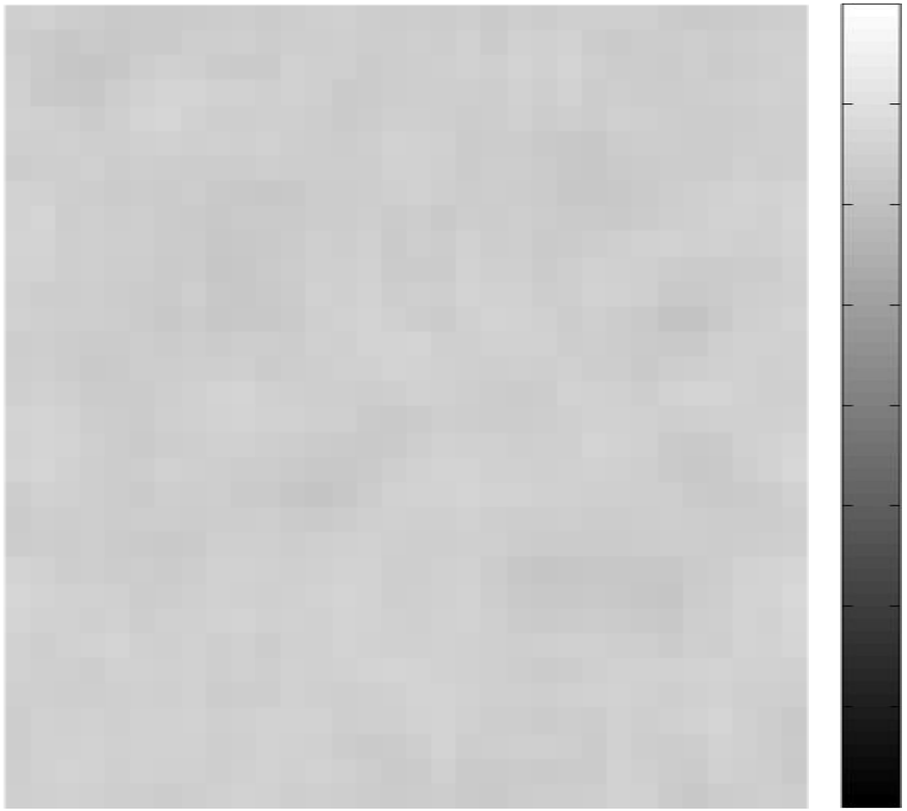
Time Series Statistics

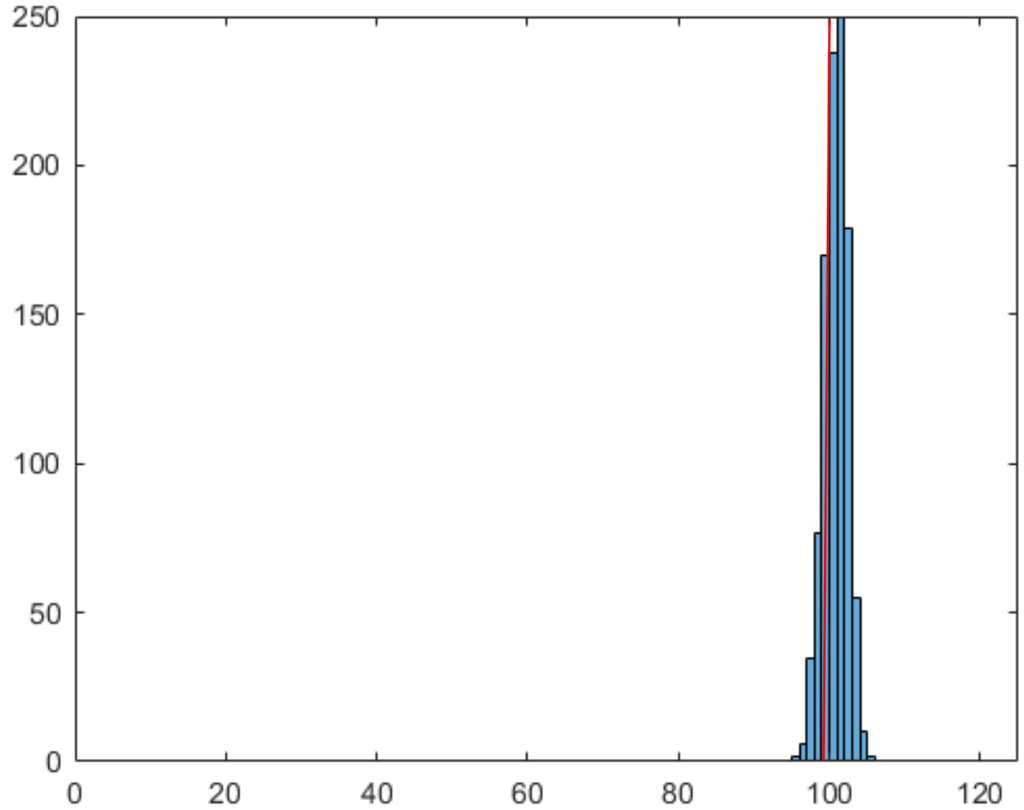
$E(x_{ji})=100$ and $var(x_{ji})=20/9$.

1	1	1
1	1	1
1	1	1

/9

If we smooth it with a 3×3 local averaging kernel,





$\bar{I}=100.8518$

$s_I^2=2.3362$

then we have kept the mean value in the image, but reduced the noise variance! (Actually also induced a local correlation!)

Time Series Statistics

```
rng('default')

nx=32; ny=nx;
mu=100; sigma2=20;

I0=ones(ny,nx);
N=sqrt(sigma2)*randn([ny,nx])+mu;
I=I0+N;

limMin=0; limMax=250; histmax=250;
figure;
imagesc(I,[limMin,limMax])
axis image, axis off, colormap(gray)

figure;
histogram(I(:),[(0:1: limMax)])
xlim([limMin,limMax]), ylim([0,histmax])

Ibar=mean(I(:))
Ivar= var(I(:))
```

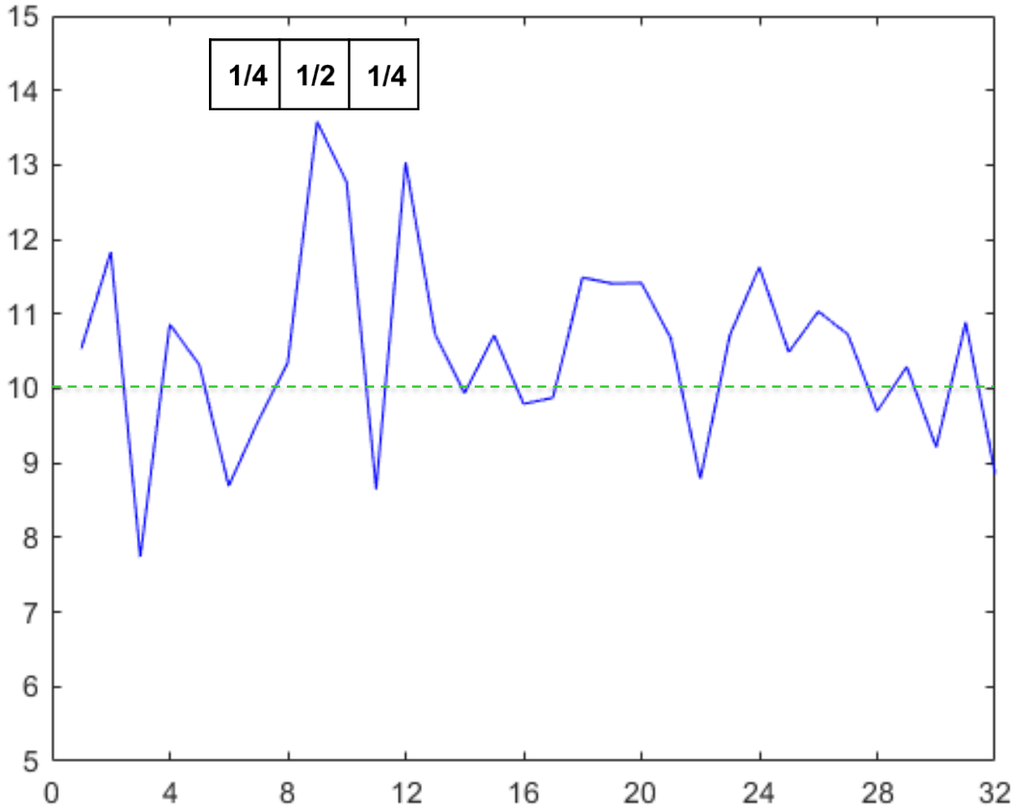
```
O1=2*I;

figure;
imagesc(O1,[limMin,limMax])
axis image, axis off, colormap(gray)

figure;
histogram(O1(:),[(0:1: limMax)])
xlim([limMin,limMax]), ylim([0,histmax])

O1bar=mean(O1(:)), O1var= var(O1(:))
```

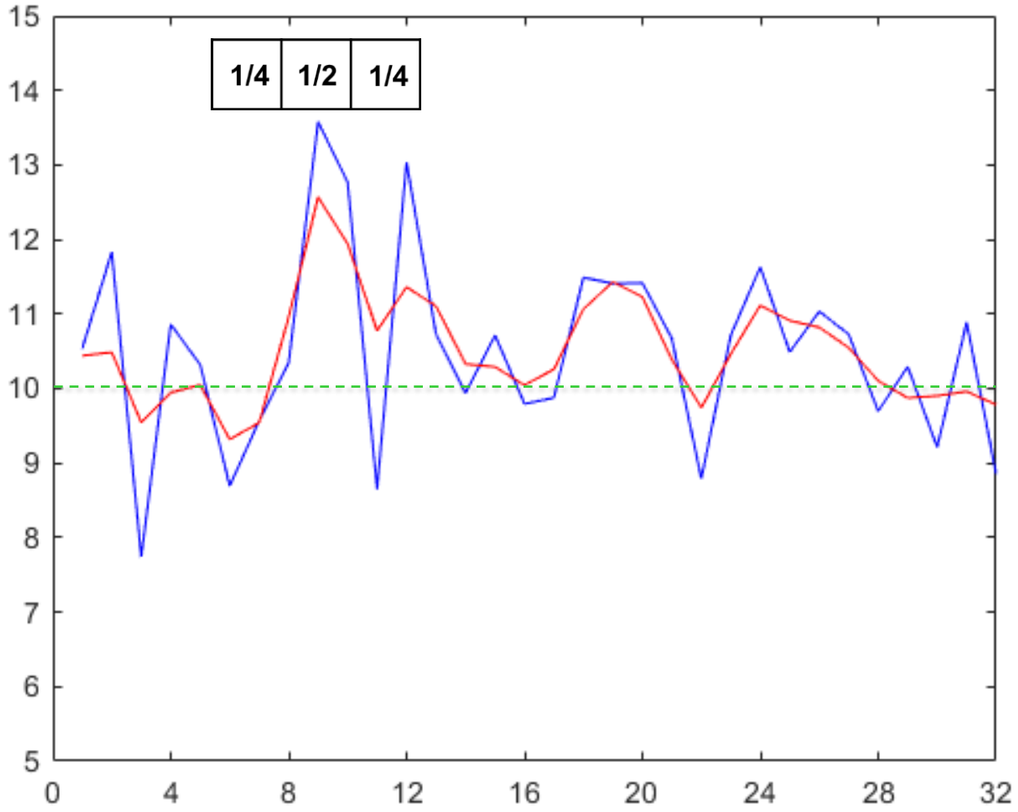
Time Series Statistics

```
limMin=0; limMax=125; histmax=250;
figure;
imagesc(I,[limMin,limMax])
axis image, axis off, colormap(gray)

figure;
histogram(I(:),[(0:1: limMax)])
xlim([limMin,limMax]), ylim([0,histmax])

kernel=ones([3,3])/9;
O2=MyConv(I,kernel);

figure;
imagesc(O2,[limMin,limMax])
axis image, axis off, colormap(gray)

figure;
histogram(O2(:),[(0:1:limMax)])
xlim([limMin,limMax]), ylim([0,histmax])

O2bar=mean(O2(:)), O2var= var(O2(:))
```

Time Series Statistics

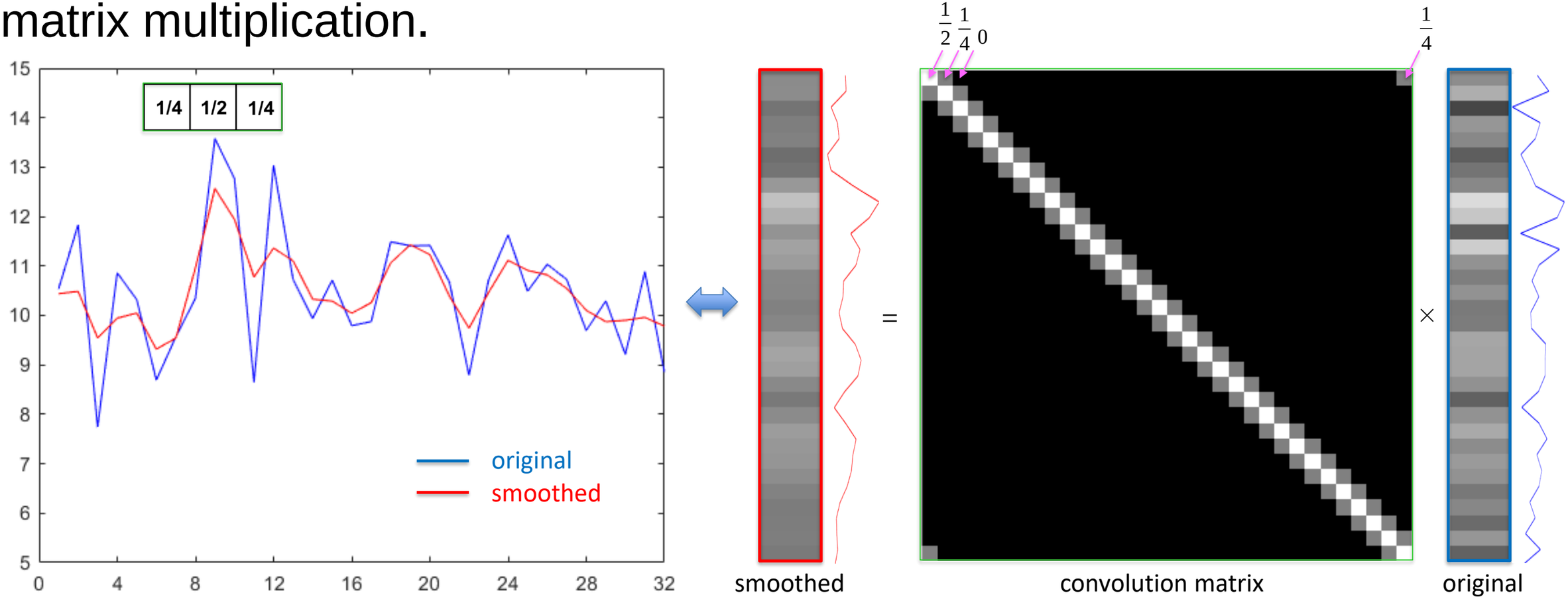
If we take a row of pixels (a time series) and smooth it with a convolution three point kernel.





Time Series Statistics

Smoothing a time series with a convolution filter can be written as a matrix multiplication.

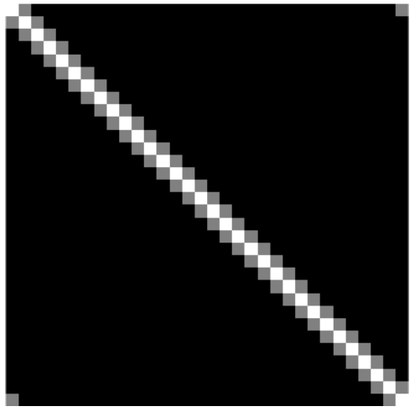


Time Series Statistics

Using the rule that we previously learned.

For this time series has a mean of $\mu=10$, variance of $\sigma^2=1$, & no correlation.

$E(Ax)=A\mu$ and $cov(Ax)= A\Sigma A'$

$A=$ 

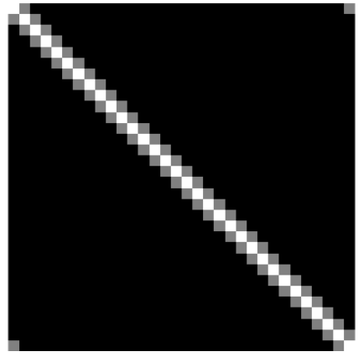
using the convolution matrix A,

$E(Ax)=10\mathbf{1}_n$ and $cov(Ax)= A(\sigma^2 \mathrm{I}_n)A'$.

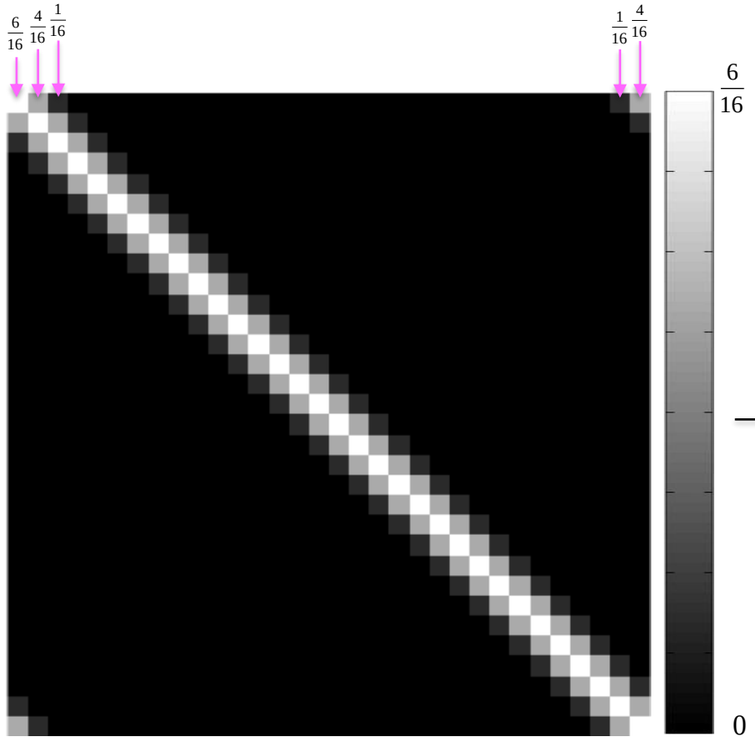
Time Series Statistics

Using the rule that we previously learned.

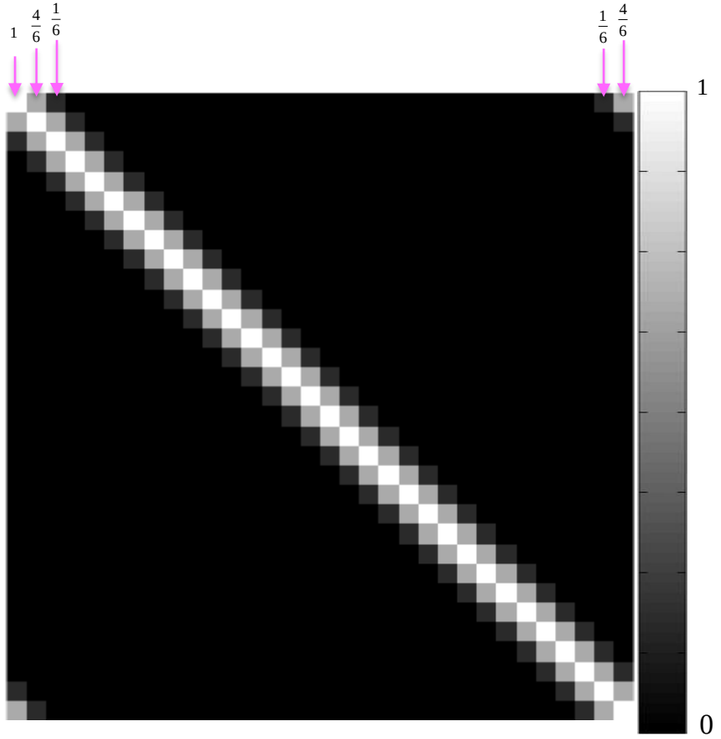
$A=$



$cov(Ax)=$

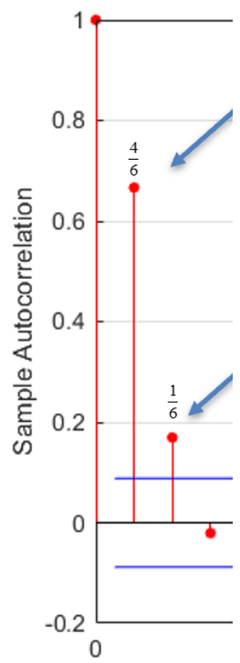


$\rightarrow$ $cor(Ax)=$



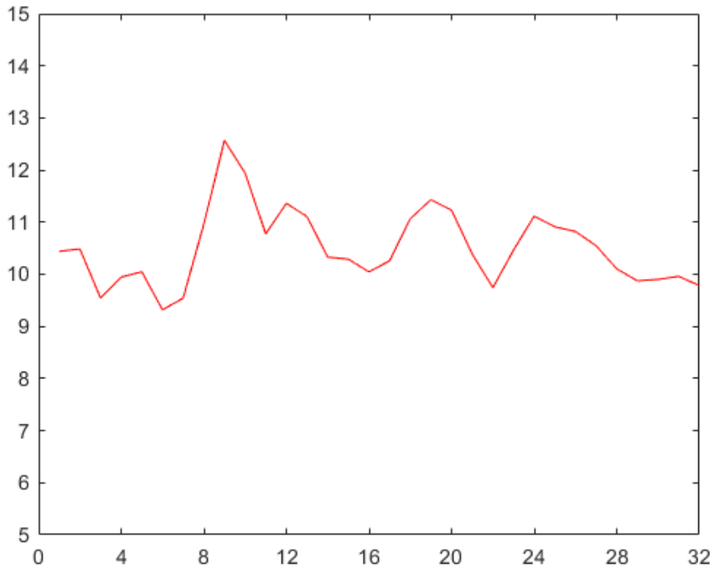
We are generally able to interpret correlation better than covariance.

recall earlier

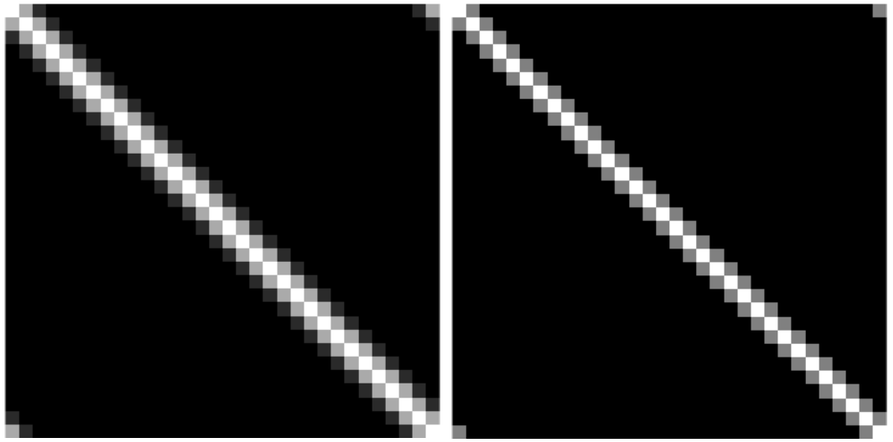


Time Series Statistics

If we assume that our time series (row of pixels) is initially uncorrelated, then if we can estimate the temporal correlation, we can deconvolve or undo the smoothing of the row of pixels.

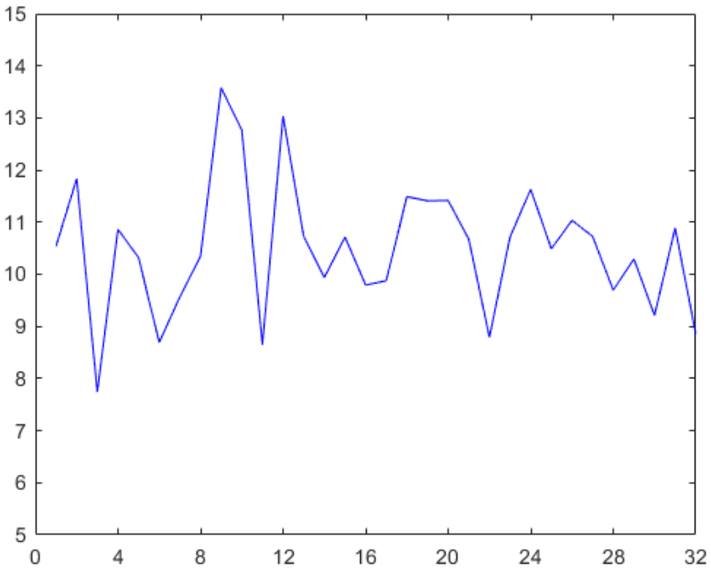


smoothed



covariance → A

deconvolution

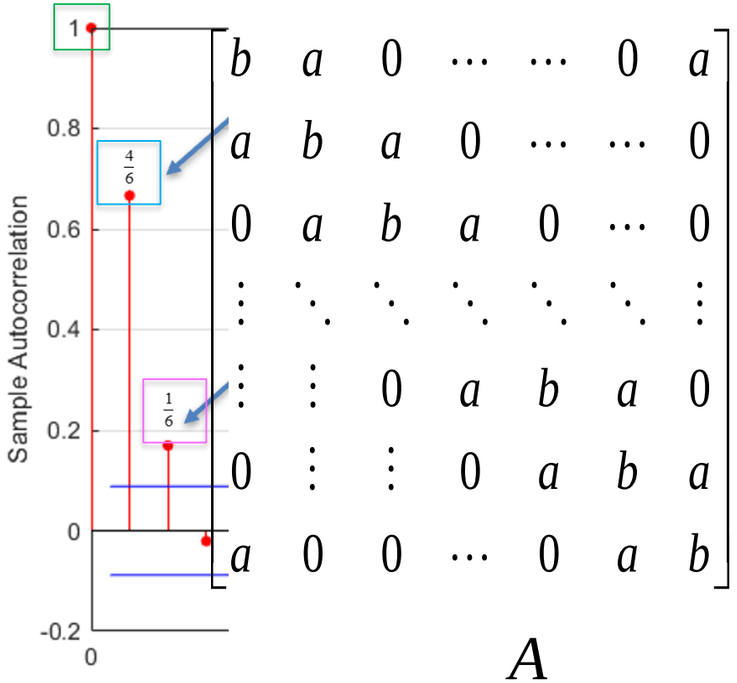


recovered

Time Series Statistics

This slide assumes $\sigma^2=1$.

Because there is a two step correlation between pixels in the time series (row), we can determine the weights in the kernel used for convolution.



$$\begin{bmatrix} b & a & 0 & \cdots & \cdots & 0 & a \\ a & b & a & 0 & \cdots & \cdots & 0 \\ 0 & a & b & a & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & 0 & a & b & a & 0 \\ 0 & \vdots & \vdots & 0 & a & b & a \\ a & 0 & 0 & \cdots & 0 & a & b \end{bmatrix} \times \begin{bmatrix} b & a & 0 & \cdots & \cdots & 0 & a \\ a & b & a & 0 & \cdots & \cdots & 0 \\ 0 & a & b & a & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & 0 & a & b & a & 0 \\ 0 & \vdots & \vdots & 0 & a & b & a \\ a & 0 & 0 & \cdots & 0 & a & b \end{bmatrix} = \begin{bmatrix} b^2+2a^2 & 2ab & a^2 & 0 & \cdots & 0 & a^2 \\ 2ab & b^2+2a^2 & 2ab & a^2 & \cdots & \cdots & 0 \\ a^2 & 2ab & b^2+2a^2 & 2ab & a^2 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & a^2 & 2ab & b^2+2a^2 & 2ab & a^2 \\ a^2 & \vdots & \vdots & a^2 & 2ab & b^2+2a^2 & 2ab \\ 2ab & a^2 & 0 & \cdots & a^2 & 2ab & b^2+2a^2 \end{bmatrix}$$

A $\qquad$ A' $\qquad$ $AA'=cov(w)$

$b^2+2a^2=1$ $\qquad$ $2ab=\frac{2}{3}$ $\qquad$ $a^2=\frac{1}{6}$ $\longrightarrow$ $a=\frac{1}{\sqrt{6}}, b=\frac{\sqrt{6}}{3}$ mult by $\frac{\sqrt{6}}{4}$ for $a=\frac{1}{4}, b=\frac{1}{2}$.

normalize

Form $B=A^{-1}$ and deconvolve!

Time Series Statistics

Unsmoothing a time series with a deconvolution filter can be written as a matrix multiplication.

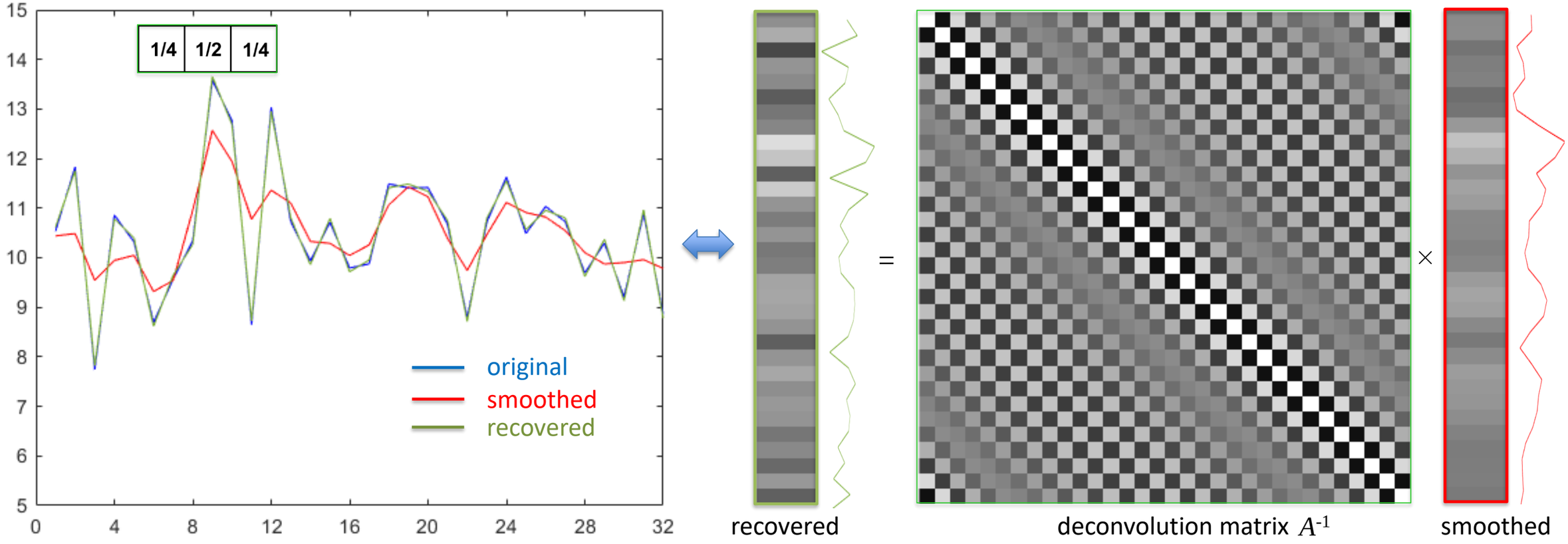


Image Statistics

If we have an image of pixels and we perform an operation on them, we can change it's statistical properties in the same way.

If we smooth an image with a convolution kernel, then we induce a local spatial correlation in the same way.

Similarly, we can estimate the local spatial correlation and deconvolve.

Image Statistics

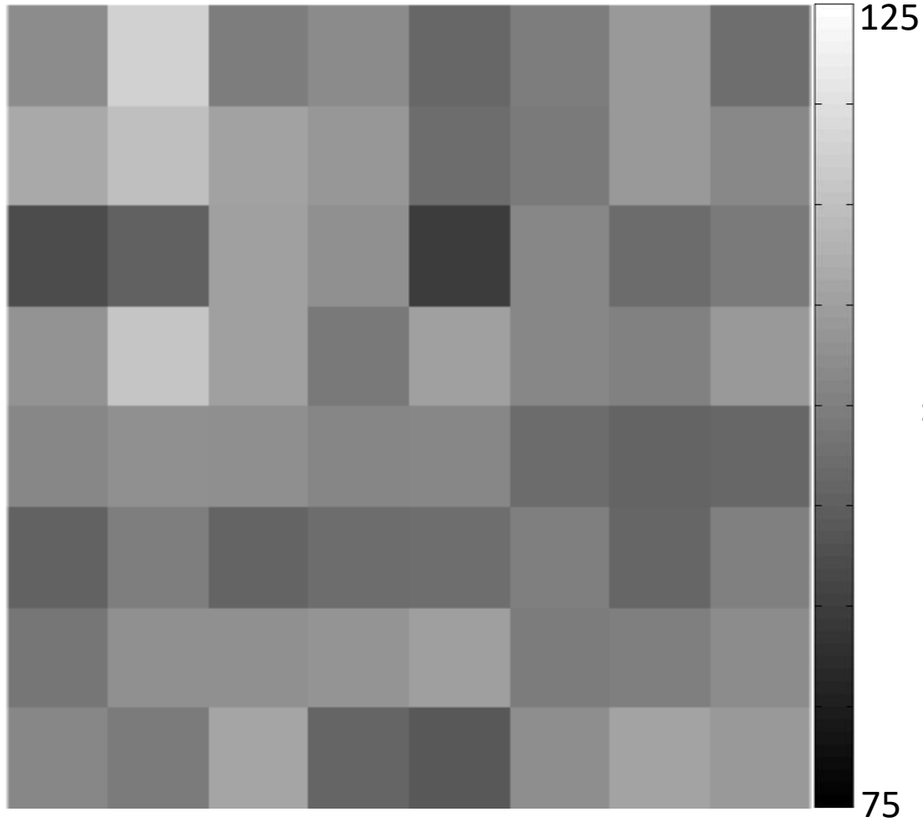
Imagine we have the below 8×8 noisy image and smooth with kernel.

0	1/8	0
1/8	1/2	1/8
0	1/8	0

Kernel

$*$

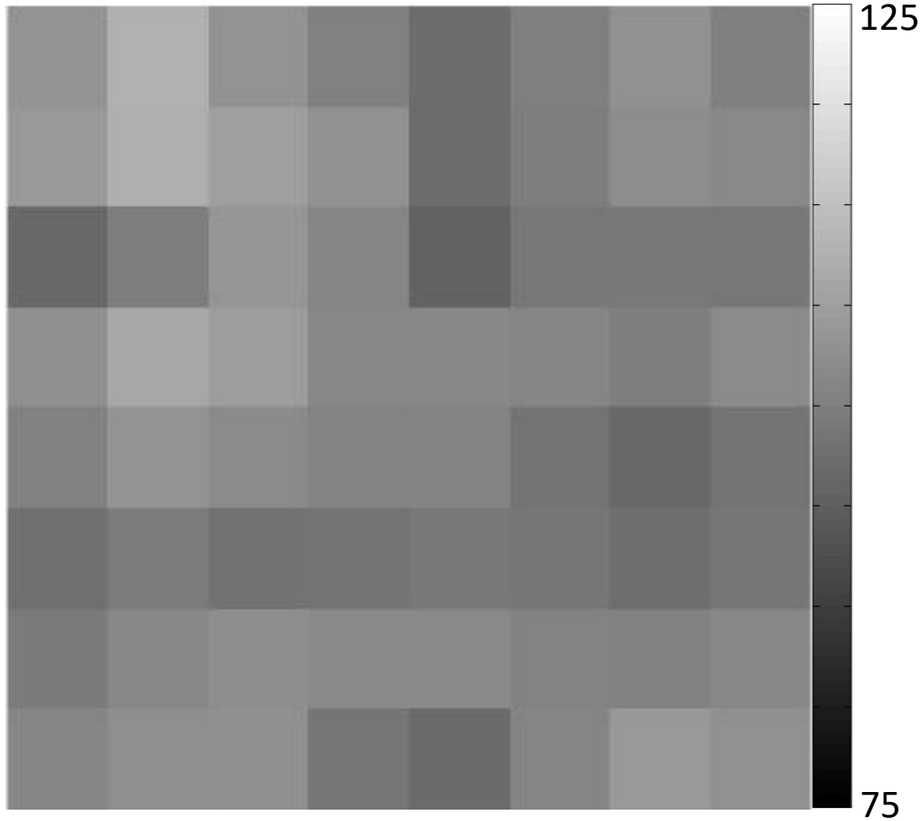
convolution



Input Image

$\bar{I} = 100.9576 \qquad s_I^2 = 28.4930$

$=$

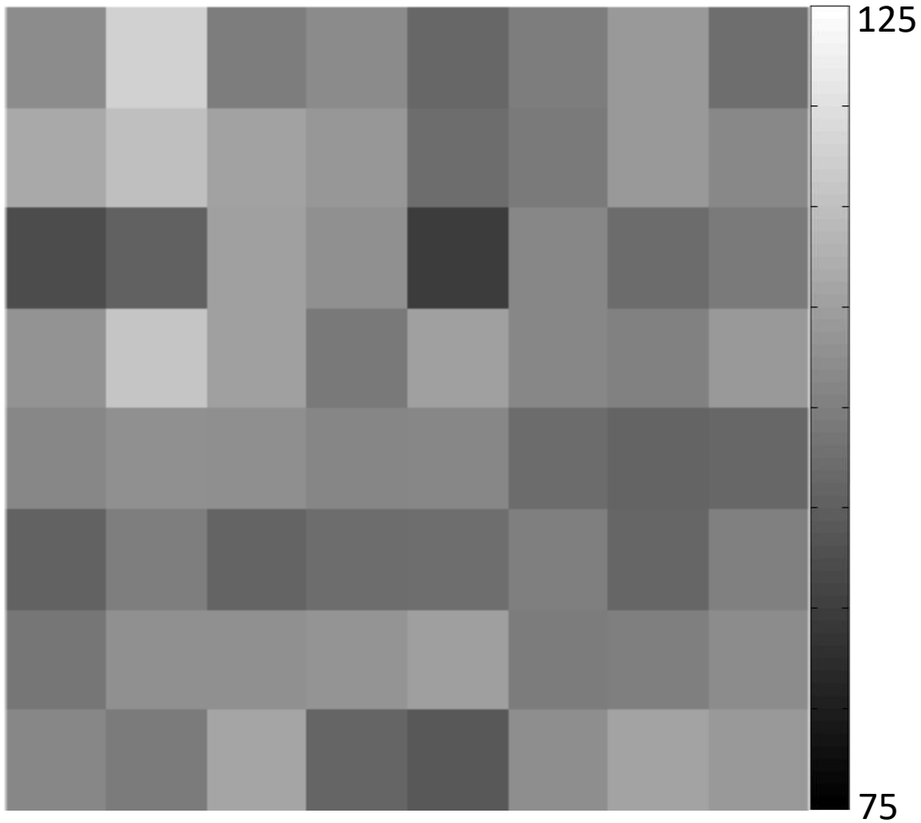


Output Image

$\bar{O} = 100.9576 \qquad s_O^2 = 10.4009$

Image Statistics

We can represent the image as a vector by stacking the rows.



Input Image

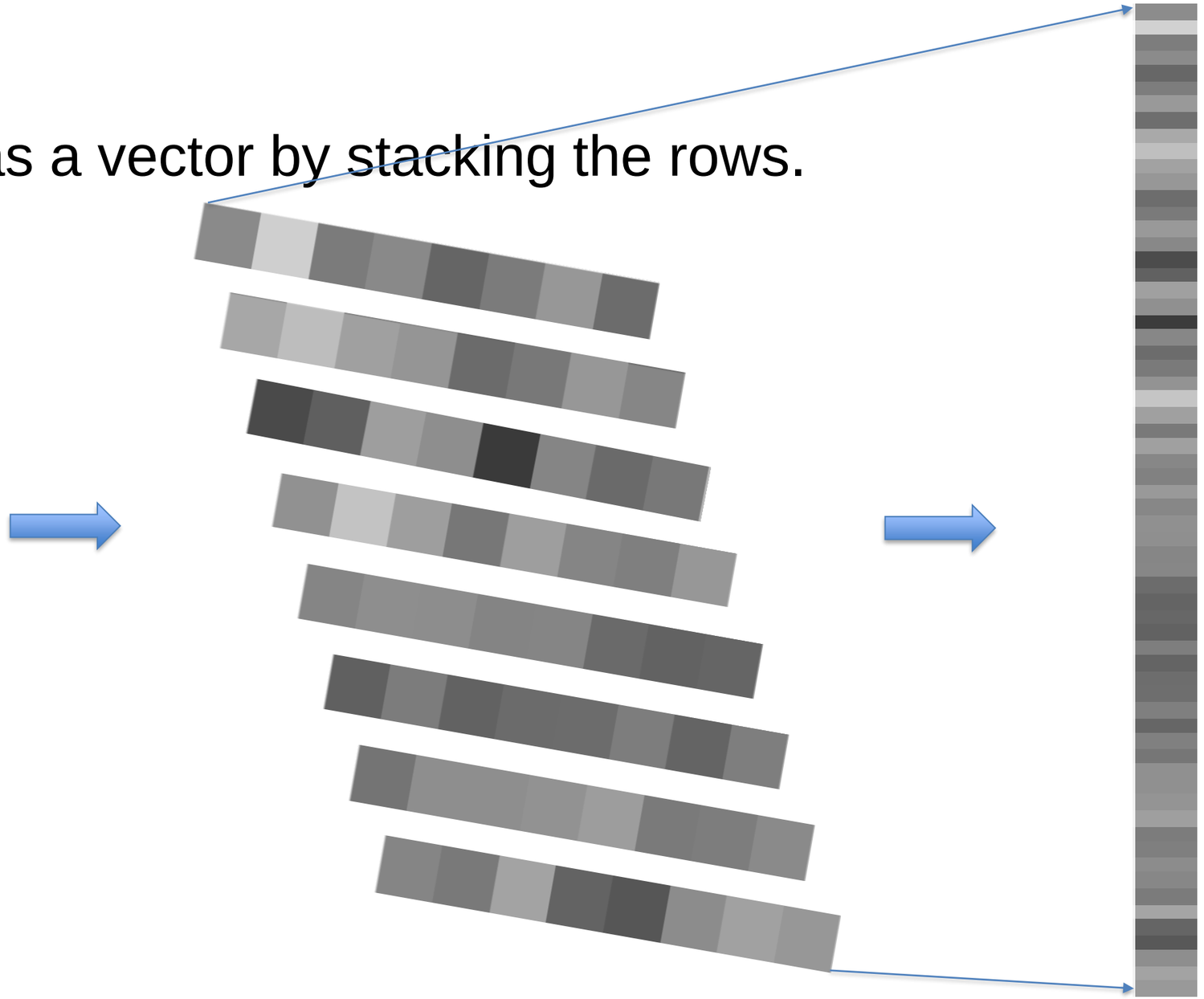


Image Statistics

0	1/8	0
1/8	1/2	1/8
0	1/8	0

Then perform convolution via a convolution matrix.

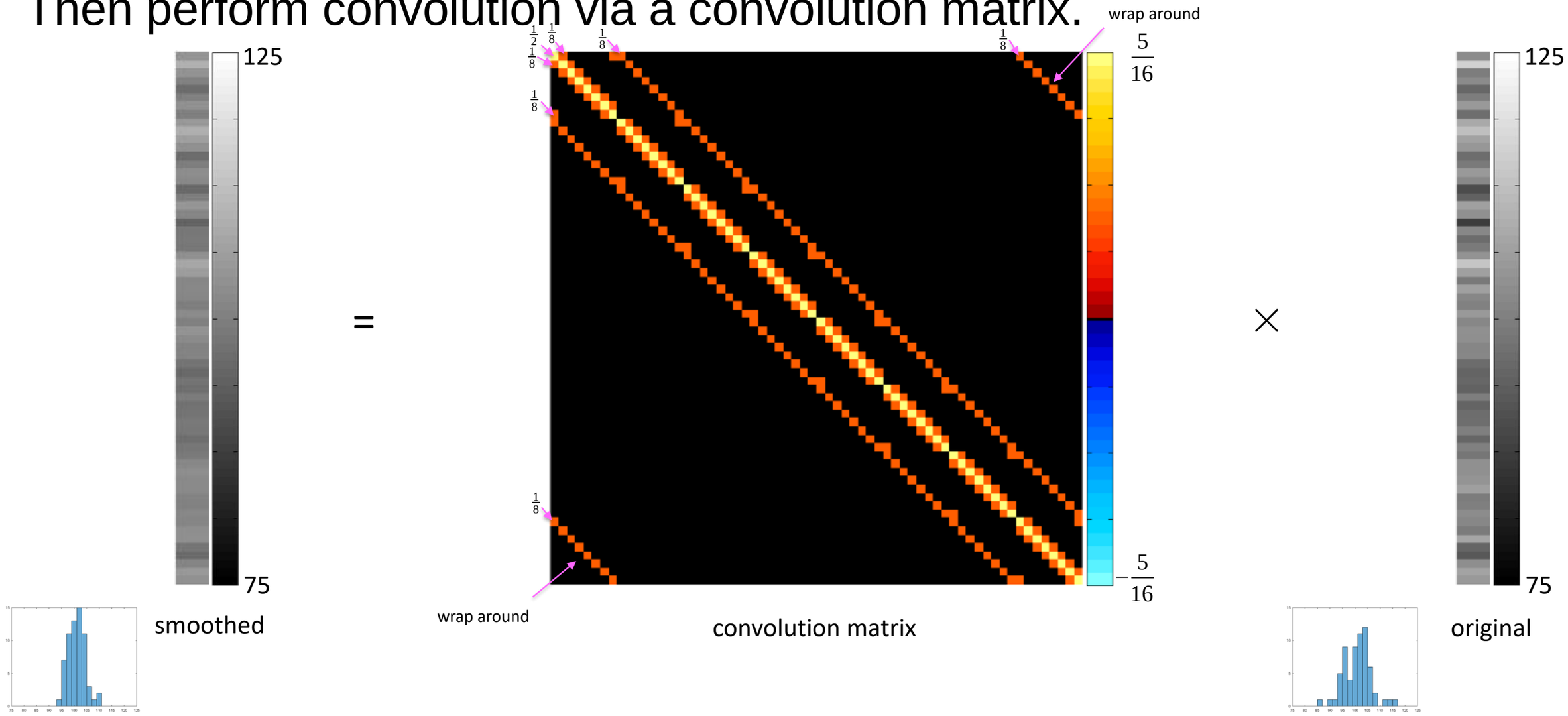


Image Statistics

$E(Ax)=A\mu$ and $cov(Ax)= A(\sigma^2 I_n)A'$.

0	1/8	0
1/8	1/2	1/8
0	1/8	0

We can calculate theoretically what the induced covariance is.

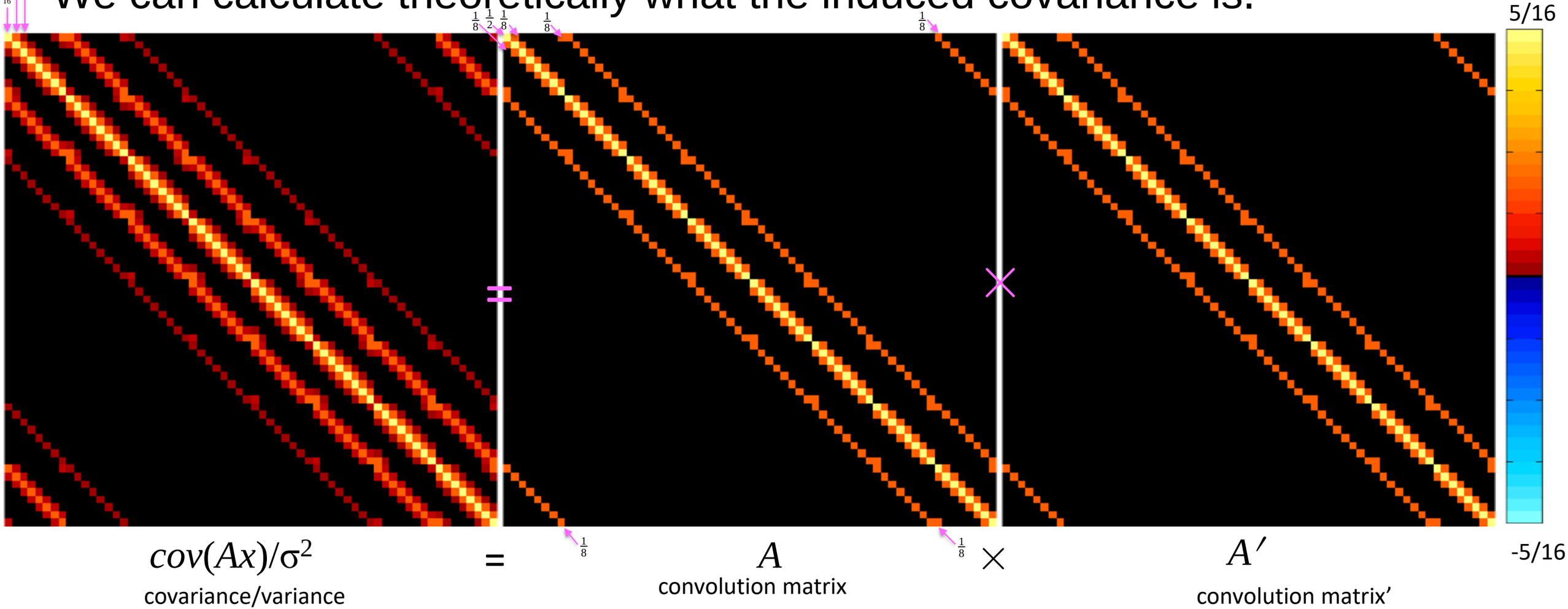
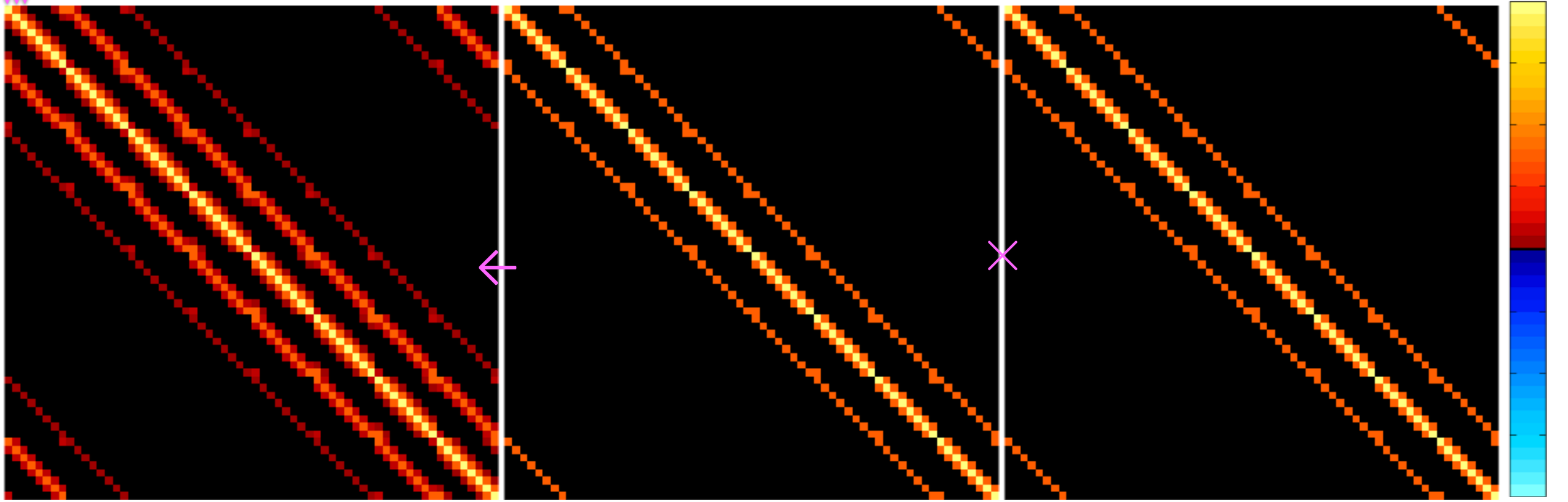


Image Statistics

0	1/8	0
1/8	1/2	1/8
0	1/8	0

We can calculate theoretically what the induced correlation is.

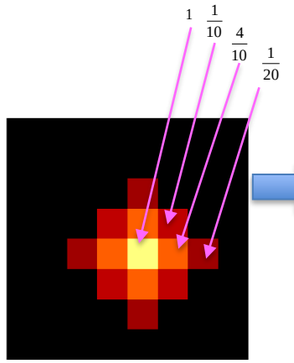


R correlation $\leftarrow$ A convolution matrix $\times$ A' convolution matrix'

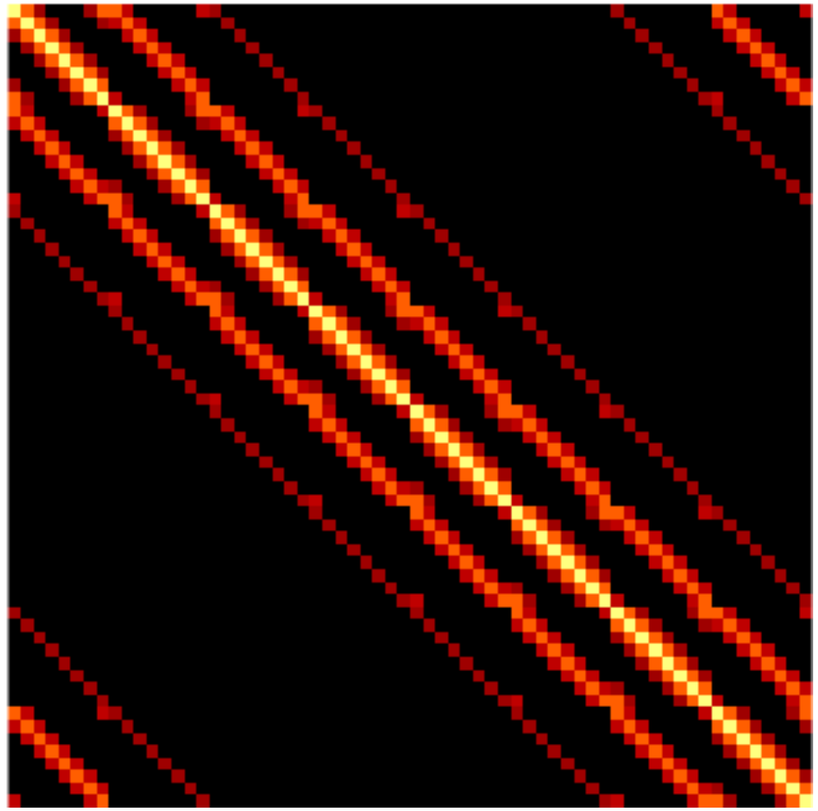
Image Statistics

0	1/8	0
1/8	1/2	1/8
0	1/8	0

If we had the induced correlation, we could calculate convolution matrix.

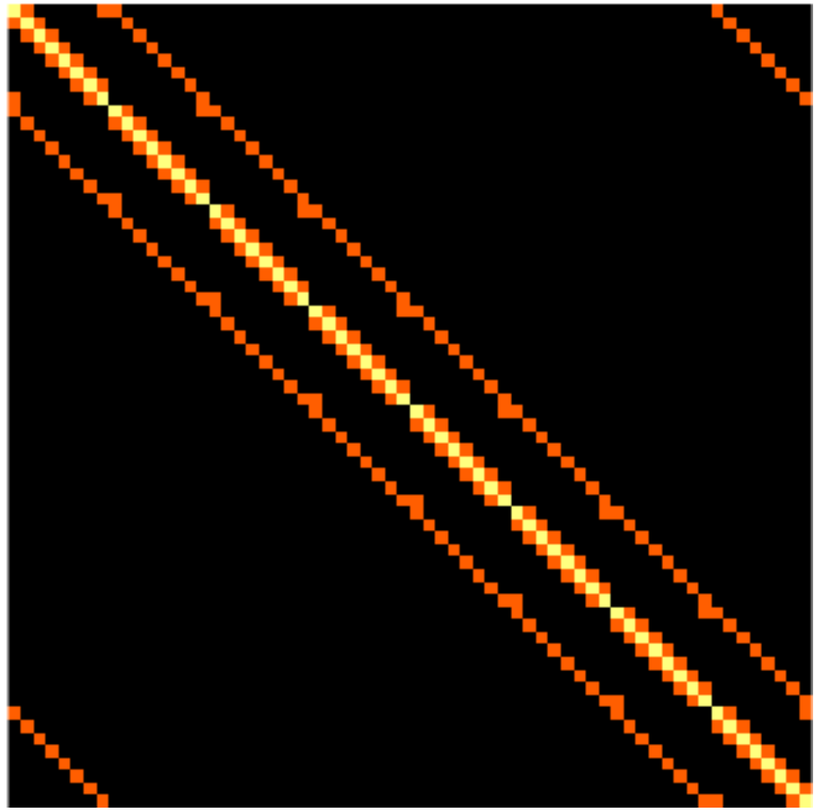


Correlation between (5,5) pixel and all others. Reshaped 37th row of R.



R

correlation



A

convolution matrix

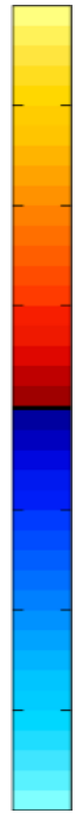
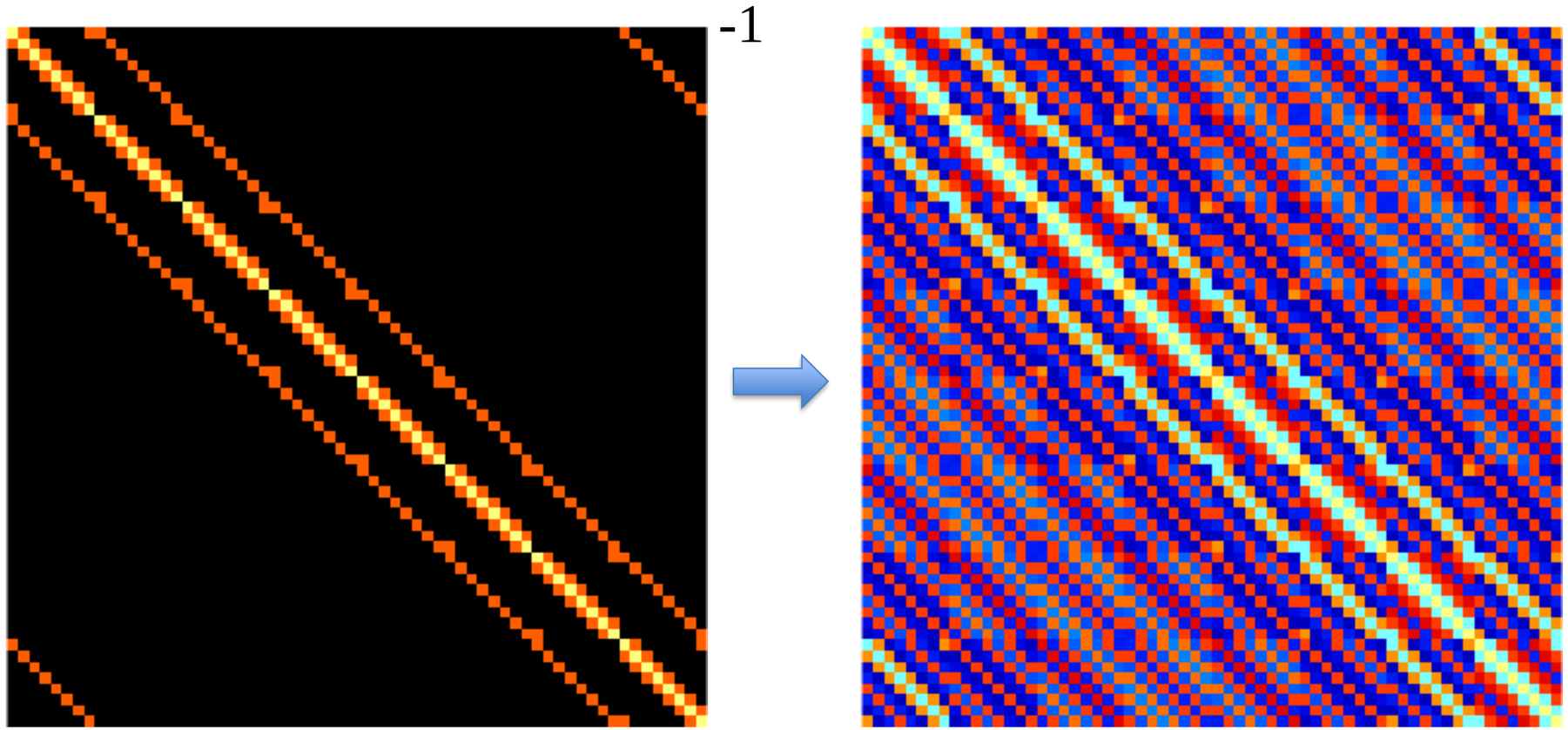


Image Statistics

0	1/8	0
1/8	1/2	1/8
0	1/8	0

Calculate inverse of convolution matrix for a deconvolution matrix.



A

convolution matrix

A^{-1}

deconvolution matrix

Image Statistics

0	1/8	0
1/8	1/2	1/8
0	1/8	0

Then perform deconvolution via a deconvolution matrix.

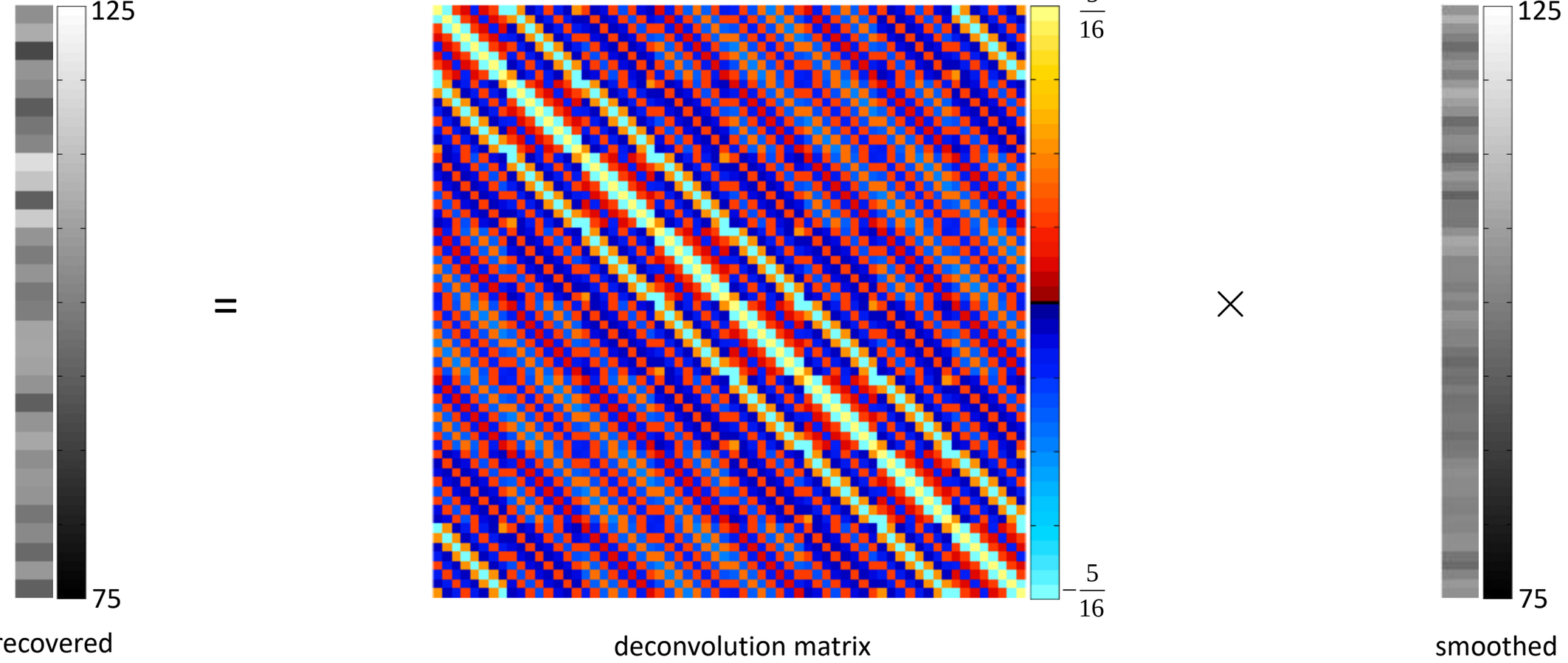
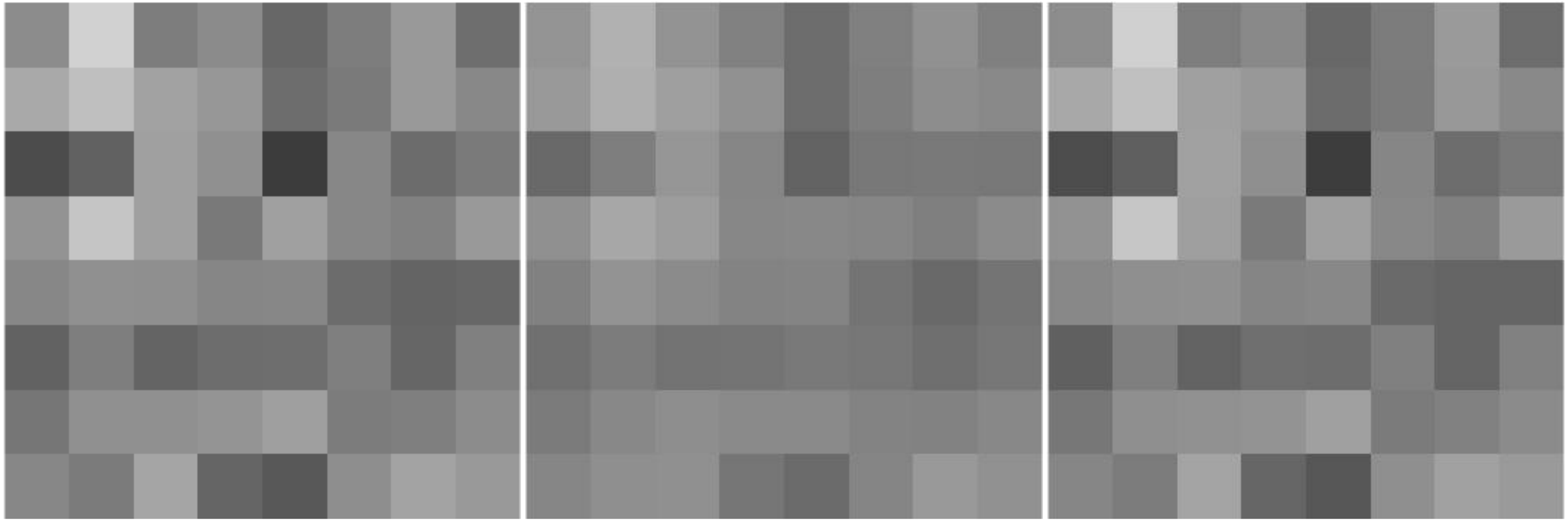


Image Statistics

Calculate inverse of convolution matrix for deconvolution matrix.



Input Image

Smoothed Image

Recovered

Image Statistics

```
rng('default')
printfigs=0;
load myposnegmapblk.txt

nx=8; ny=nx;
limMin=75; limMax=125; yhist=15;

mu=100; sigma2=20;
N=sqrt(sigma2)*randn([ny,nx]);
I=mu+N;

Ibar=mean(I(:)), s2I=var(I(:))

figure;
imagesc(I,[limMin,limMax])
colormap(gray), axis image, axis off
figure;
histogram(I(:),(limMin:2:limMax))
xlim([limMin,limMax]), ylim([0,yhist])
figure;
imagesc(repmat(reshape(I',nx*ny,1),[1,4]),[limMin,limMax])
colormap(gray), axis image, axis off
```

```
kernel=[  0,1/8,  0;...
        1/8,1/2,1/8;...
          0,1/8,  0];
O=MyConv(I,kernel);

figure;
imagesc(O,[limMin,limMax])
colormap(gray), axis image, axis off
figure;
histogram(O(:),(limMin:2:limMax))
xlim([limMin,limMax]), ylim([0,yhist])
figure;
imagesc(repmat(reshape(O',nx*ny,1),[1,4]),...
    [limMin,limMax])
colormap(gray), axis image, axis off
Obar=mean(O(:)), s2O=var(O(:))
krnl=[1/2,1/8,  0,  0,  0,  0,  0,1/8;...
      1/8,  0,  0,  0,  0,  0,  0,  0;...
        0,  0,  0,  0,  0,  0,  0,  0;...
        0,  0,  0,  0,  0,  0,  0,  0;...
        0,  0,  0,  0,  0,  0,  0,  0;...
        0,  0,  0,  0,  0,  0,  0,  0;...
        0,  0,  0,  0,  0,  0,  0,  0;...
      1/8,  0,  0,  0,  0,  0,  0,  0];
```

Image Statistics

```
A=zeros(ny*nx,ny*nx);
for j=1:ny
    if (j~=1)
        krnl=[krnl(ny,:);krnl(1:ny-1,:)];
    end
    A((j-1)*nx+1,:)=reshape(krnl',[nx*ny,1])';
    for i=(j-1)*nx+2:j*nx
        krnl=[krnl(:,nx),krnl(:,1:nx-1)];
        A(i,:)=reshape(krnl',[nx*ny,1])';
    end
    krnl=[krnl(:,nx),krnl(:,1:nx-1)];
end
figure;
imagesc(A,[0,5/16])
axis image, axis off, colormap(gray)
figure;
imagesc(A,[-5/16,5/16])
axis image, axis off
colormap(myposnegmapblk)
AAt=A*A';
figure;
imagesc(AAt,[0,5/16])
colormap(gray), axis image, axis off
```

```
R=diag(1./sqrt(diag(AAt)))*AAt*diag(1./sqrt(diag(AAt)));
figure;
imagesc(R,[0,1])
colormap(gray), axis image, axis off
figure;
imagesc(R,[-1,1])
colormap(myposnegmapblk), axis image, axis off
Ivec=reshape(I',nx*ny,1);
Wvec=A*Ivec;
W=reshape(Wvec,nx,ny)';
figure;
imagesc(W,[limMin,limMax])
colormap(gray), axis image, axis off
figure;
histogram(W(:),(limMin:2:limMax))
xlim([limMin,limMax]), ylim([0,yhist])
figure;
imagesc(repmat(reshape(W',nx*ny,1),[1,4]),[limMin,limMax])
colormap(gray), axis image, axis off
Wbar=mean(W(:)), s2W=var(W(:))
```

Image Statistics

```
invA=pinv(A);
figure;
histogram(invA(:))
figure;
imagesc(invA,[-1,1])
colormap(gray), axis image, axis off
figure;
histogram(invA(:))
figure;
imagesc(invA,[-1,1])
colormap(myposnegmapblk), axis image, axis off
Vvec=invA*Wvec;
V=reshape(Vvec,nx,ny)';
figure;
imagesc(V,[limMin,limMax])
colormap(gray), axis image, axis off
figure;
histogram(V(:),(limMin:2:limMax))
xlim([limMin,limMax]), ylim([0,yhist])
figure;
imagesc(repmat(reshape(V',nx*ny,1),[1,4]),[limMin,limMax])
colormap(gray), axis image, axis off
Vbar=mean(V(:)), s2V=var(V(:))
```

```
pixnum=ny/2*nx+nx/2+1;
r=R(pixnum,:);
corImage=reshape(r,[nx,ny])';
figure;
imagesc(corImage,[-1,1])
axis image, axis off , colormap(myposnegmapblk)
```

Image Statistics

```
% Perform image space convolution with wrap around
% usage is O=MyConv(I,kernel)
% I=input image, O=output image, k=linear kernel

function O=MyConv(I,kernel);

[a,b]=size(kernel);
[n,m]=size(I);
% appends border pixels for wrap-around
IW=[I(n-(a-1)/2+1:n,m-(b-1)/2+1:m),I(n-(a-1)/2+1:n,1:m),I(n-(a-1)/2+1:n,1:(b-1)/2);...
    I(1:n          ,m-(b-1)/2+1:m),I(1:n          ,1:m),I(1:n          ,1:(b-1)/2);...
    I(1:(a-1)/2    ,m-(b-1)/2+1:m),I(1:(a-1)/2    ,1:m),I(1:(a-1)/2    ,1:(b-1)/2)];
O=zeros(n+(a-1),m+(b-1));
for j=1+(a-1)/2:n+(a-1)/2
    for i=1+(b-1)/2:m+(b-1)/2
        patch =IW(j-(a-1)/2:j+(a-1)/2,i-(b-1)/2:i+(b-1)/2);
        O(j,i)=sum(sum(patch.*kernel));
    end
end
% remove appended pixels
O(1:(a-1)/2,:)=[];
O(n+1:n+(a-1)/2,:)=[];
O(:,1:(b-1)/2)=[];
O(:,m+1:m+(b-1)/2)=[];
end
```

Image Statistics

To calculate the spatial correlation, we first subtract the mean value of each pixel from its pixel value for residual. Constant image easy.

Then need to loop through each pixel.
Keep track of pixel residual and residual for pixels 1 away, $\sqrt{2}$ away, 2 away,… Each pixel has 4 of 1 away neighbors, 4 of $\sqrt{2}$ away neighbors, and 4 of 2 away neighbors, … there are $4n_x n_y$ of each type.

After forming a list of pixels and neighbors, calculate spatial correlation.

$$\begin{array}{ccccc} & & 2 & & \\ & \sqrt{2} & 1 & \sqrt{2} & \\ 2 & 1 & 0 & 1 & 2 \\ & \sqrt{2} & 1 & \sqrt{2} & \\ & & 2 & & \end{array}$$

Euclidean Distance

$$\begin{array}{ccccc} & & 3 & & \\ & 2 & 1 & 2 & \\ 3 & 1 & 0 & 1 & 3 \\ & 2 & 1 & 2 & \\ & & 3 & & \end{array}$$

Distance Type

Discussion

Whenever we do something to data, we change it's statistics.

If a convolution kernel has been applied to a noisy image where the noise between pixels was originally uncorrelated, the pixels now have an induced local correlation.

If we can estimate the local correlation, then we can deconvolve the filtered image and recover the original image.

Discussion

Questions?

Homework 4

1. Generate 10^6 random observations from a normal distribution with mean 100 and variance 4. Calculate mean and variance, make a histogram. Multiply the observations by 7. Calculate mean, variance, make histogram.

2. Generate a time series of observations of length n=200. Calculate sample mean, variance, and make a histogram. Apply your own 3 point kernel convolution to it. Calculate sample mean, variance, and make a histogram.

3*.Calculate the autocorrelation in #2. Deconvolve the time series. Write your own Matlab code to deconvolve. No Matlab functions.

*For students in MSSC 5770.

Homework 4

4. Make a constant image and add normally distributed noise.
Smooth your image. Calculate mean and variance of all pixels in the image and make a histogram. Do both before and after.

5**.Calculate the spatial autocorrelation in #4. Deconvolve the image.
Write your own Matlab code, no Matlab functions.

**For students in MSSC 5770 that want to demonstrate how smart they are.

Submit one document with all your results (no Matlab code) and some discussion of thoughts or commentary.
Separately also submit executable Matlab code and any needed files.