from here: http://nipunbatra.github.io/2014/05/denoising/ (http://nipunbatra.github.io/2014/05/denoising/)

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn
%matplotlib inline
```

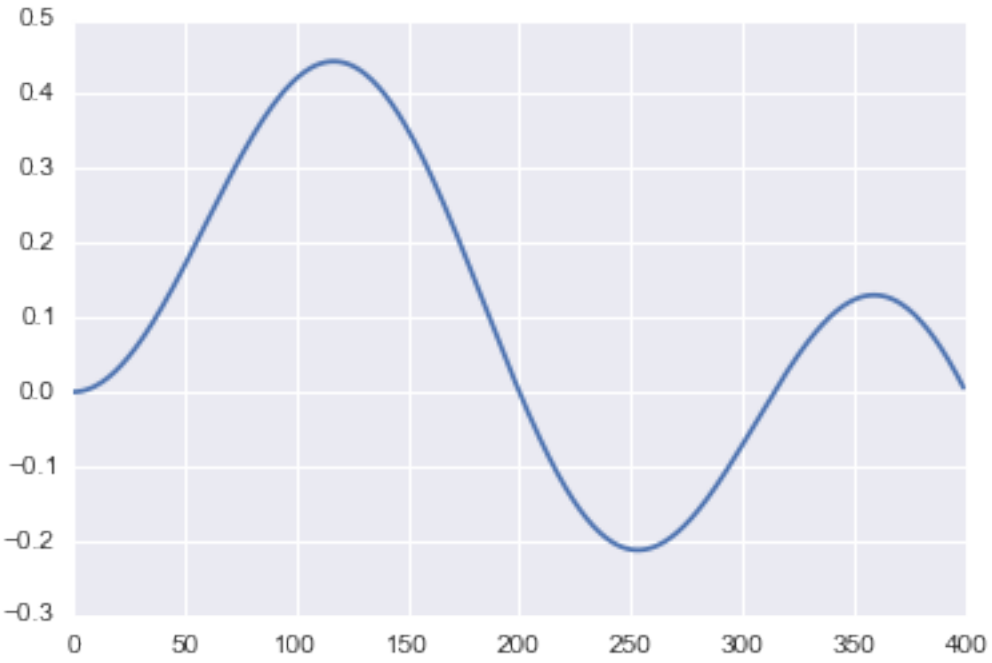
True Data

In [2]:

```
n=400
t=np.array(range(n))
ex=0.5*np.sin(np.dot(2*np.pi/n,t))*np.sin(0.01*t)
plt.plot(ex)
```

Out[2]:

```
[<matplotlib.lines.Line2D at 0x10f87f4a8>]
```



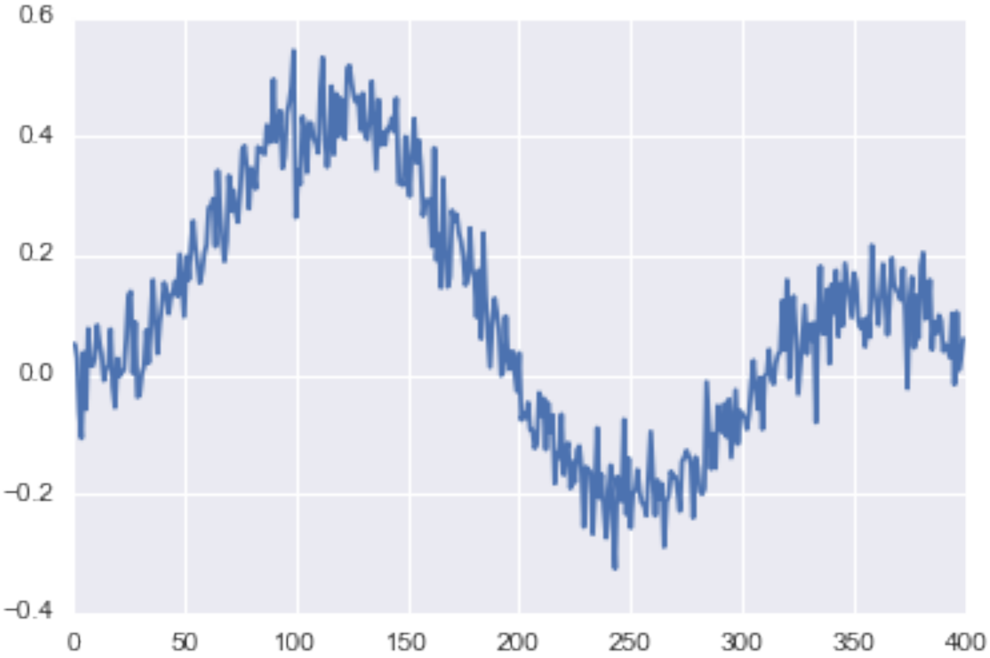
Noisy Data

In [3]:

```
corr=ex+0.05*np.random.normal(0,1,n)
plt.plot(corr)
```

Out[3]:

```
[<matplotlib.lines.Line2D at 0x10f9a1240>]
```



What do we want?

1. We want to clean the data so that it's "close" to the noisy data
2. We want the cleaned data to be "smooth"

Constraint 1

We really want

$$\|x - x_{corr}\|^2$$

to be minimized.

In terms of least-squarse ($Ax = b$) this means that

$$A_{first} = I \qquad b_{first} = x_{corr}$$

Constraint 2

For the smooth data we can ask that

$$\mu \sum_{k=1}^{n-1} (x_{k+1} - x_k)^2$$

is minimized

In terms of least-squares, this means that

$$A_{second} = \begin{bmatrix} -1 & 1 & 0 & \dots \\ 0 & -1 & 1 & \dots \\ \dots & 0 & -1 & 1 \end{bmatrix}$$

and

$$b = 0$$

In [9]:

```
d1=np.eye(n-1,n)
d1=np.roll(d1,1)
print(d1)
```

```
[[ 0.  1.  0. ...,  0.  0.  0.]
 [ 0.  0.  1. ...,  0.  0.  0.]
 [ 0.  0.  0. ...,  0.  0.  0.]
 ..., 
 [ 0.  0.  0. ...,  1.  0.  0.]
 [ 0.  0.  0. ...,  0.  1.  0.]
 [ 0.  0.  0. ...,  0.  0.  1.]]
```

In [30]:

```
root_mu=100
d2=-np.eye(n-1,n)
a_second=root_mu*(d1+d2)
print(a_second)
print(a_second.shape)
```

```
[[-100.  100.    0. ...,    0.    0.    0.]
 [   0. -100.  100. ...,    0.    0.    0.]
 [   0.    0. -100. ...,    0.    0.    0.]
 ..., 
 [   0.    0.    0. ...,  100.    0.    0.]
 [   0.    0.    0. ..., -100.  100.    0.]
 [   0.    0.    0. ...,    0. -100.  100.]]
(399, 400)
```

In [31]:

```
a_first=np.eye(n)
A=np.vstack((a_first,a_second))
```

In [32]:

```
corr=corr.reshape((n,1))
b_2=np.zeros((n-1,1))
b=np.vstack((corr,b_2))
```

In [33]:

```
print(A.shape)
print(b.shape)
```

```
(799, 400)
(799, 1)
```

Now solve the least squares problem

In [34]:

```
import numpy.linalg.linalg as la
```

In [35]:

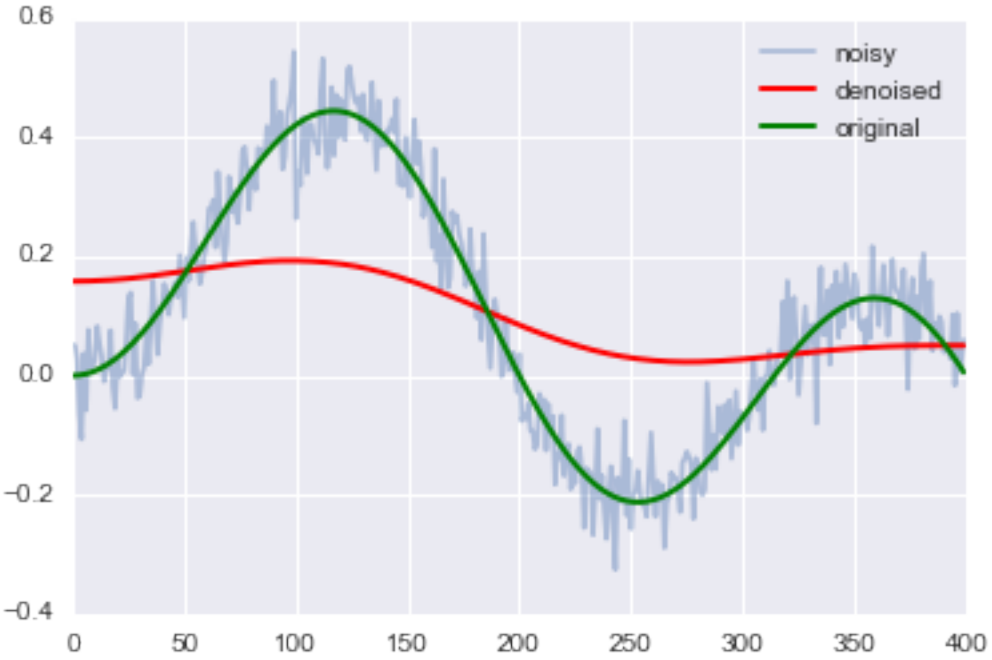
```
sol = la.lstsq(A, b)
```

In [36]:

```
plt.plot(corr,label='noisy',alpha=0.4)
plt.plot(sol[0],'r',linewidth=2, label='denoised')
plt.plot(ex,'g',linewidth=2, label='original')
plt.legend()
```

Out[36]:

```
<matplotlib.legend.Legend at 0x11174b278>
```



In [ ]: