Error Propagation

Normal distribution

```
In [1]: import numpy as np
        import matplotlib.pyplot as plt
```

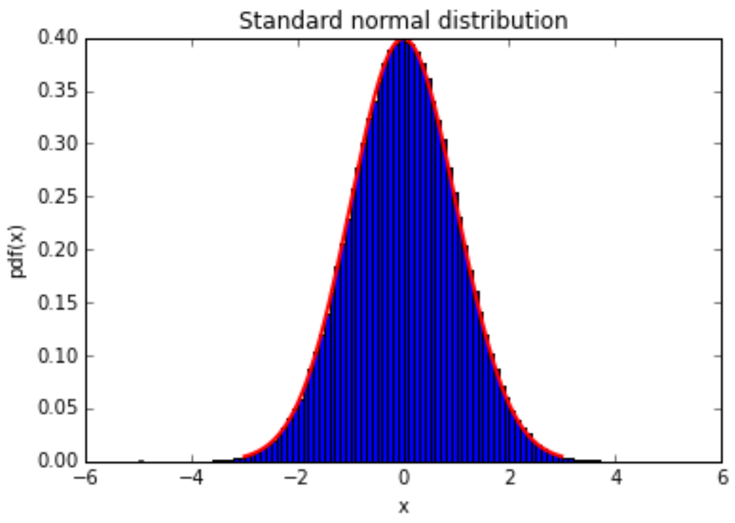
We generate random numbers drawn from a standard $(\mu = 0, \sigma = 1)$ normal distribution, i.e. from

$$\mathrm{prob}(x|\mu = 0, \sigma = 1) = \mathrm{pdf}(x|\mu = 0, \sigma = 1)dx = \frac{dx}{\sqrt{2\pi}}\exp\left(-\frac{x^2}{2}\right).$$

```
In [2]: rns = np.random.randn(1000000)
```

These numbers can be plotted in a histogram for which we specify the number $n = 100$ of bins and the range $[-5, 5]$. The option normed=True normalizes the bin heigths such that the integral of the histogram is one. Note that if you specify the range for the histogram, it will be normalized within this range. This makes it necessary to choose the range large enough to cover the full distribution including the tails.

```
In [3]: plt.figure(1)
        plt.hist(rns,bins=100,range=[-5,5],normed=True);
        x=np.arange(-3,3,0.01);
        y=1/np.sqrt(2*np.pi)*np.exp(-x**2/2)
        plt.plot(x,y,linewidth=2,color='red')
        plt.title('Standard normal distribution')
        plt.xlabel('x')
        plt.ylabel('pdf(x)')
        plt.show()
```



Suppose we are now interested in the distribution of

$$y = \sigma x + \mu.$$

In general, we find such a distribution $g(y)$ from the distribution $f(x)$ by requiring

$$f(x)|dx| = g(y)|dy|.$$

Step 1: calculate the total differential dy. We have

$$dy = \sigma dx.$$

Step 2: insert it into the equation.

$$f(x)dx = g(y)\sigma dx \quad \Rightarrow \quad g(y) = \frac{1}{\sigma}f(x).$$

Step 3: insert $x(y)$ on the right hand side. Here we have

$$x = \frac{y-\mu}{\sigma}.$$

Note that in general, the transformation function must have an inverse. This gives

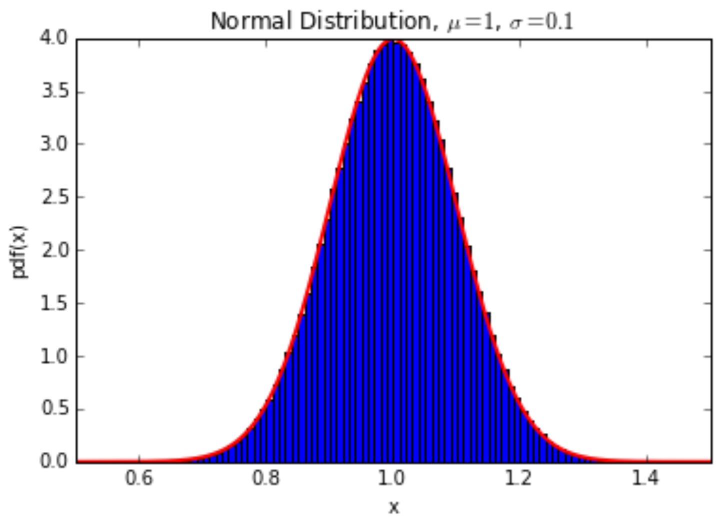
$$g(y) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2\right).$$

In order to test our result, we apply the transformation rule

$$x \to y = \sigma x + \mu$$

to the random numbers chosing $\sigma = 0.1$ and $\mu = 1$, and plot the resulting histogram together with the calculated probability density function $g(y)$.

```
In [4]: plt.figure(1)
        plt.hist(0.1*rns+1,bins=100,range=(0.5,1.5),normed=True);
        x=np.arange(0.5,1.5,0.001);
        y=1/np.sqrt(2*np.pi*0.01)*np.exp(-(x-1)**2/2/0.01)
        plt.plot(x,y,linewidth=2,color='red')
        plt.axis([0.5,1.5,0,4])
        plt.title(r'Normal Distribution, $\mu=1$, $\sigma=0.1$')
        plt.xlabel('x')
        plt.ylabel('pdf(x)')
        plt.show()
```



Exponential decay on a logarithmic scale

We apply the same coordinate transformation technique to another problem that commonly occurs in the lab. Suppose you have measured a quantity $x(t)$ which is supposed to follow the exponential law

$$x(t) = \exp(-\gamma t).$$

The goal is to determine γ from the experimentally determined data (t_i, x_i). You assume that the x_i have an addititve error, which is normally distributed. Your colleague suggests that you plot the measured values x on a logarithmic scale and then do standard linear regression for determining γ. The new variable is therefore

$$y = \ln(x),$$

which gives the linear relation

$$y = -\gamma t.$$

What is the distribution of the errors in y?

We answer this question by transforming the normal distribution to the logarithmic scale.

Step 1:

$$dy = \frac{dx}{x}.$$

Step 2:

$$f(x)dx = g(y)\frac{dx}{|x|} \quad \Rightarrow \quad g(y) = |x|f(x).$$

Step 3:

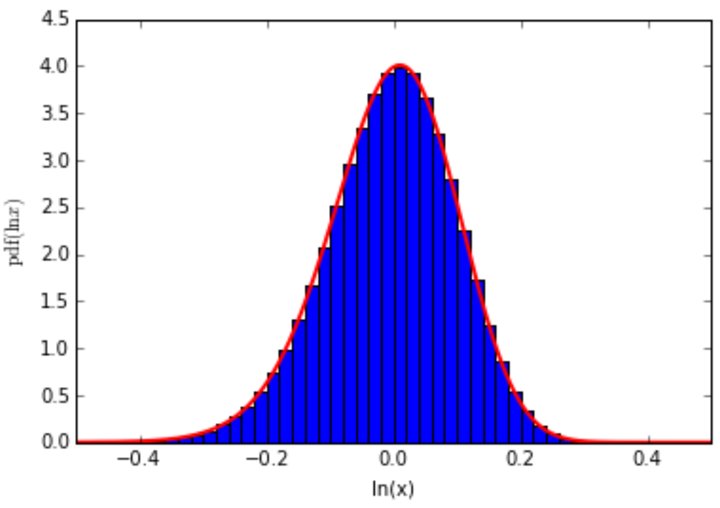
$$x = \exp(y) \quad \Rightarrow \quad g(y) = \exp(y)f[\exp(y)].$$

Inserting the normal distribution, we find

$$g(y) = \exp(y)\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(\exp(y)-\mu)^2}{2\sigma^2}\right)$$

Again we compare this analytic result with the histogram representation of the transformed random samples.

```
In [5]: plt.figure(1)
        plt.hist(np.log(0.1*rns+1),bins=100,range=(-1,1),normed=True);
        x=np.arange(-1,1,0.001);
        y=np.exp(x)/np.sqrt(2*np.pi*0.01) * np.exp(-(np.exp(x)-1)**2/2/0.01)
        plt.plot(x,y,linewidth=2,color='red')
        plt.axis([-0.5,0.5,0,4.5])
        plt.xlabel('ln(x)')
        plt.ylabel(r'$\mathrm{pdf}(\ln x)$')
        plt.show()
```



We see that the resulting distribution is not a normal distribution. Furthermore, the normal distribution is not a good approximation because of the asymmetry. This implies that ordinary linear regression cannot be applied to the problem on the logarithmic scale! You better do non-linear data fitting of the exponential function within the framework of gaussian error analysis.

Reciprocal normal distribution

Suppose we perform an experiment in which an object travels with constant velocity along a length $L = 1\,\mathrm{m}$. We measure the velocity v of the object with a newly developed speedometer and wish to determine the travel time from

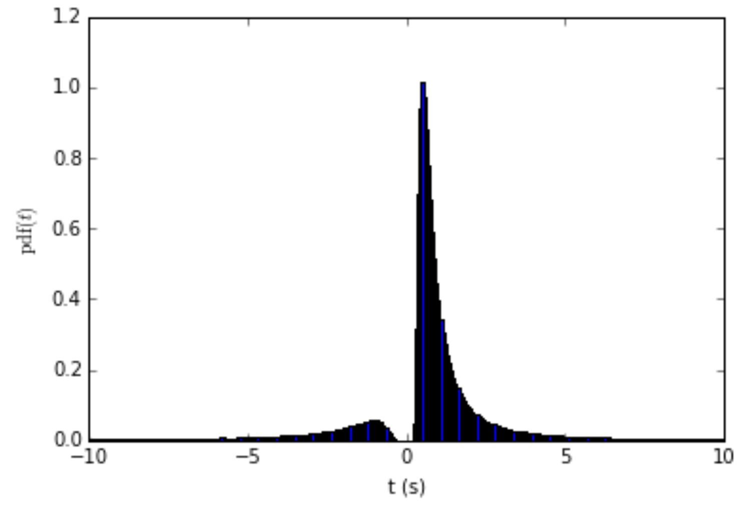
$$t = \frac{L}{v}.$$

The speedometer has an uncertainty of $1\,\mathrm{m/s}$. In a particular experiment the speedometer reads the value $v = 1\,\mathrm{m/s}$. What is the uncertainty of the travel time?

Here we are interested in the distribution of $y = 1/(x+1)$, where x is normally distributed with mean $\mu = 0$ and standard deviation $\sigma = 1$. The distribution of y can be found from our random numbers:

```
In [6]: plt.hist(1/(rns+1),bins=300,range=[-10,10],normed=True,stacked=True);
        plt.xlabel('t (s)')
        plt.ylabel(r'$\mathrm{pdf}(t)$')
```

```
Out[6]: <matplotlib.text.Text at 0x1122d4990>
```



We see that the distribution $g(z)$ has a quite complicated shape with a suppressed probability around zero, and long tails towards large positive and negative values. Furthermore, the distribution has two maxima. It can be shown that this distribution does neither have a mean nor a variance.

We find the analytic form of this distribution using the transformation

$$x \to y = 1/x.$$

Step 1: Calculate the differential dy. We have

$$dy = -\frac{dx}{x^2}.$$

Step 2: Insert it into the equation

$$f(x)|dx| = g(y)\frac{|dx|}{x^2} \qquad \Rightarrow \qquad g(y) = x^2 f(x).$$

Step 3: replace x by $x(y) = 1/y$. We find

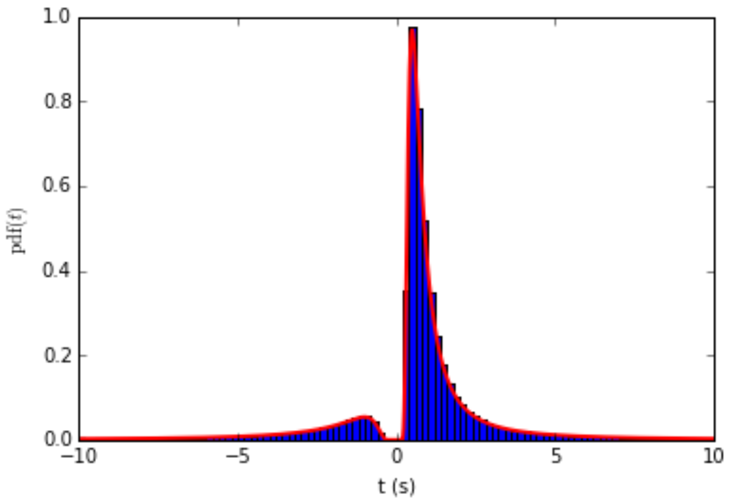
$$g(y) = \frac{1}{y^2} f(1/y).$$

Inserting the normal distribution, we find

$$g(y) = \frac{1}{y^2\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\mu^2(y - 1/\mu)^2}{2\sigma^2 y^2}\right).$$

We again check this result against the histogram of the transformed random numbers.

In [7]:
```
x = np.arange(-10,10,0.01)
g = np.exp(-(x - 1)**2 / (2*x*x)) / (np.sqrt(2*np.pi) * x*x)
plt.figure(1)
plt.plot(x,g,color='red',linewidth=2);
plt.hist(1/(rns+1),bins=100,range=[-10,10],normed=True,stacked=True);
plt.xlabel('t (s)')
plt.ylabel(r'$\mathrm{pdf}(t)$')
plt.show()
```



Dissipated power

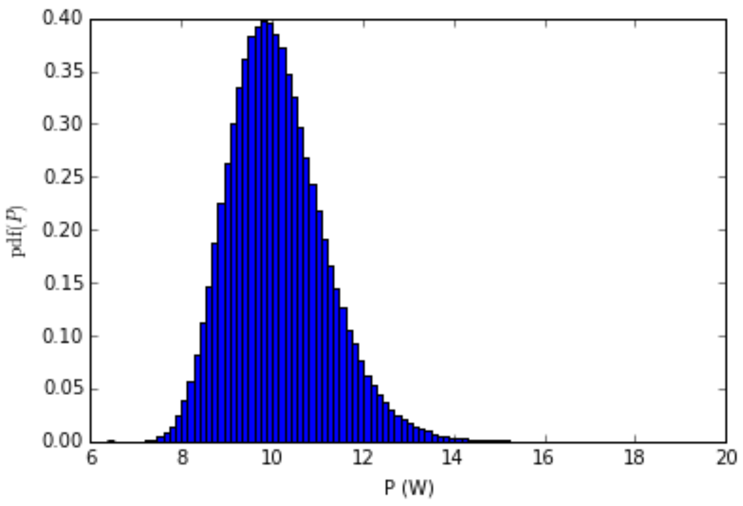
We drive an unknown current through a resistor R, which was determined in a calibration measurement to be $R = (10 \pm 1)\,\Omega$. The measured voltage drop across the resistor is $U = (10 \pm 0.1)\,V$. Estimate the power

$$P = \frac{U^2}{R}$$

dissipated in the resistor.

We solve this problem entirely using the histogram method:

In [8]:
```
rns1 = np.random.randn(1000000)
rns2 = np.random.randn(1000000)
randomU = rns1*0.1 + 10;
randomR = rns2*1 + 10;
plt.figure(1)
plt.hist(randomU*randomU/randomR,bins=100,normed=True,stacked=True);
plt.xlabel('P (W)')
plt.ylabel(r'$\mathrm{pdf}(P)$')
plt.show()
```

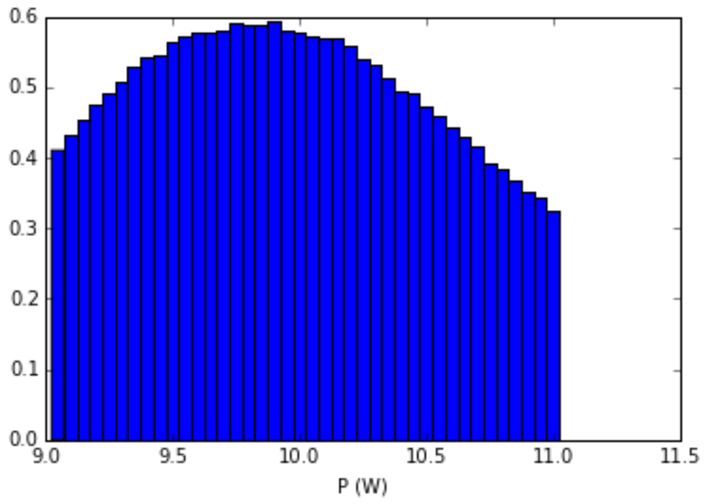


From the histogram we find the maximum of the distribution, and the width, where the distribution has reduced to $\exp(-1/2)$ of its peak value. We see that the distribution is not symmetric. The error in positive direction from the maximum will therefore be larger than the error in negative direction.

We show how a detailed inspection of the histogram can be done:

Step 1: determine the maximum. For this purpose we blow up the region around the maximum and make sure that we have a bin-width that is easy to determine. In our case we choose a bin-width of 0.1.

```
In [9]: plt.figure(1)
        plt.hist(randomU*randomU/randomR,bins=40,normed=True,range=(9.025,11.025)
        ,stacked=True);
        plt.xlabel('P (W)')
        plt.show()
```



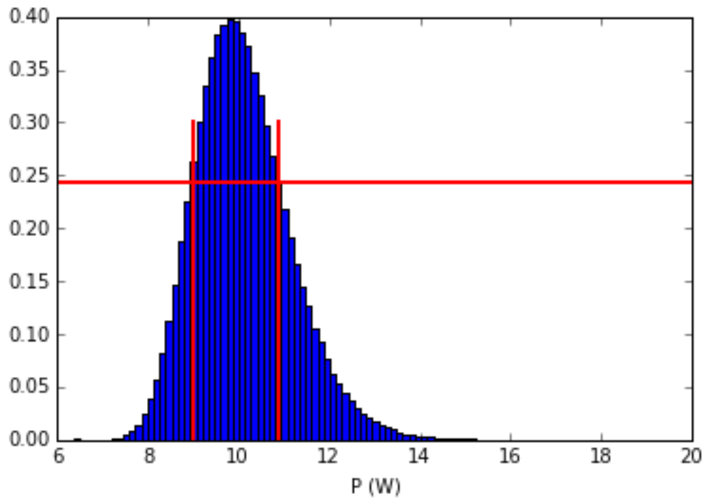
We read $P = 9.85\,\mathrm{W}$.

The width of the distribution has to be taken at

```
In [10]: 0.4*np.exp(-1/2.)

Out[10]: 0.24261226388505339
```

```
In [11]: plt.figure(1)
         plt.hist(randomU*randomU/randomR,bins=100,normed=True,stacked=True);
         plt.plot([6,20],[0.4*np.exp(-1/2.),0.4*np.exp(-1/2.)],color='red',linewid
         th=2)
         plt.plot([9,9],[0,0.3],color='red',linewidth=2)
         plt.plot([10.9,10.9],[0,0.3],color='red',linewidth=2)
         plt.xlabel('P (W)')
         plt.show()
```



```
In [12]: 10.9-9.85

Out[12]: 1.0500000000000007
```

```
In [13]: 9.85-9.

Out[13]: 0.8499999999999996
```

Our estimate is therefore $P = (9.85 + 1.05/ - 0.85)$ W.