full_calibration_analysis_example

Unknown Author

March 22, 2014

1 Calibration and uncertainty analysis - virtual experiment

In [52]:
```
from IPython.display import Image
im = Image('../Lectures/Lecture2/pressure_calibration_table.png')
im
```

Out [52]:

True pressure = 10.000 ± .001 kPa
Acceleration = 0
Vibration level = 0
Ambient temperature = 20 ± 1°C

Trial number	Scale reading, kPa
1	10.02
2	10.20
3	10.26
4	10.20
5	10.22
6	10.13
7	9.97
8	10.12
9	10.09
10	9.90
11	10.05
12	10.17
13	10.42
14	10.21
15	10.23
16	10.11
17	9.98
18	10.10
19	10.04
20	9.81

In [53]:
```
ptrue = 10.000
p = np.array([10.02, 10.20, 10.26, 10.20, 10.22, 10.13, 9.97, 10.12, 10.09, 9.9, 10.05
```

Let's build histogram, we need to select the number of bins or Δp

1.1 Recommendations for choice of the histogram size:

k or number of bins shall be at least 5:

$k \geq 5$

There are several different methods to estimate the right number of bins for the histogram:

$K = 1.87(N-1)^{0.4} + 1$

or

$K = N^{1/2}$

histogram is defined as: $Z = \frac{n(y)}{N \Delta y}$

where Δy bin size, N total number of readings, $n(y)$ is the number of readings in some bin, centered at y

In [54]:
```
K = 1.87*(p.size - 1)**(0.4); print K
K = sqrt(p.size); print K
K = 9
dp = (max(p) - min(p))/K; print dp
bins = r_[min(p)-dp:max(p)+dp:dp]; print bins # row vector
```

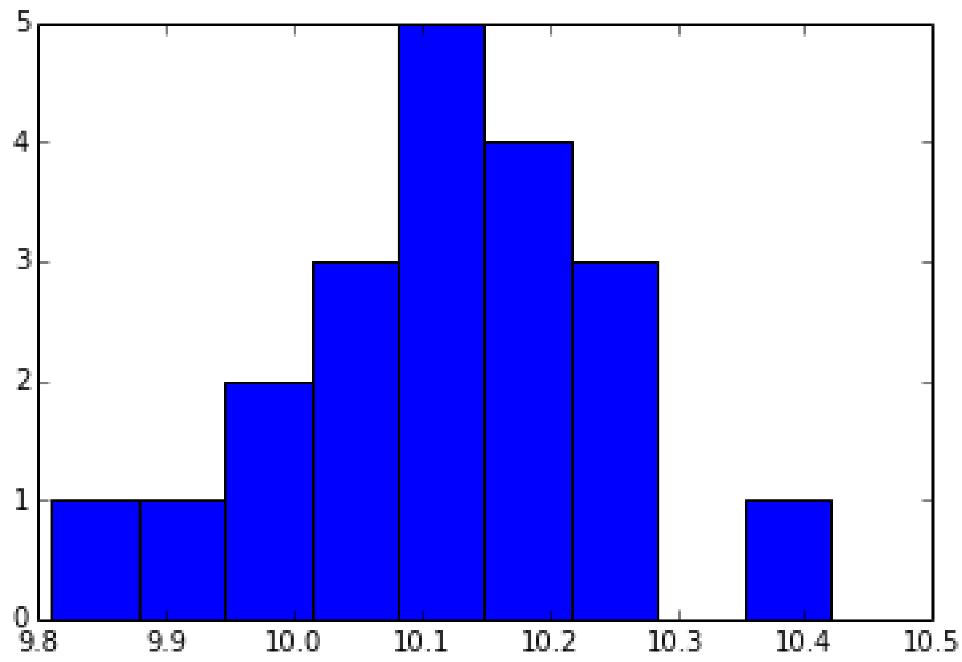
```
6.07215776742
4.472135955
0.0677777777778
[  9.74222222   9.81         9.87777778   9.94555556  10.01333333
  10.08111111  10.14888889  10.21666667  10.28444444  10.35222222
 10.42       ]
```

In [55]:
```
hist, bin_edges = np.histogram(p,bins=bins)
hist
```

Out [55]:
```
array([0, 1, 1, 2, 3, 5, 4, 3, 0, 1])
```

In [56]:
```
bar(bin_edges[:-1], hist, dp)
```

Out [56]:
```
<Container object of 10 artists>
```



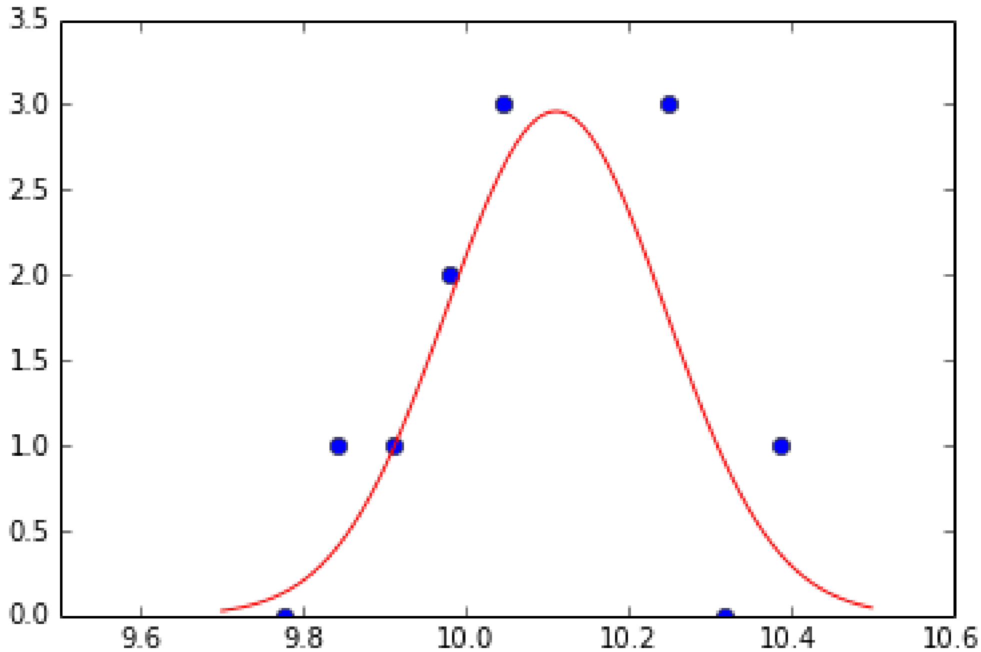
we expect to see the Gaussian, if our pressure measurements contain random errors

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(X-\mu)^2}{2\sigma^2}}$$

In [57]:
```
mu = mean(p)
sigma = std(p)
print 'mean, std %3.2f, %3.2f' % (mu, sigma)
x = linspace(9.7,10.5,100)
gauss = 1./np.sqrt(2*np.pi*sigma**2)*exp(-((x-mu)**2/(2*sigma**2)))
```
```
mean, std 10.11, 0.13
```

In [58]:
```
plot(bin_edges[:-1]+dp/2,hist,'bo',x,gauss,'r')
ylim(0,3.5)
xlim(9.5, 10.6)
```
```
(9.5, 10.6)
```

Out [58]:



χ^2 test

How do we check if our histogram is similar to the Gaussian (or any other) distribution? Goodness-of-fit is called the χ^2 test

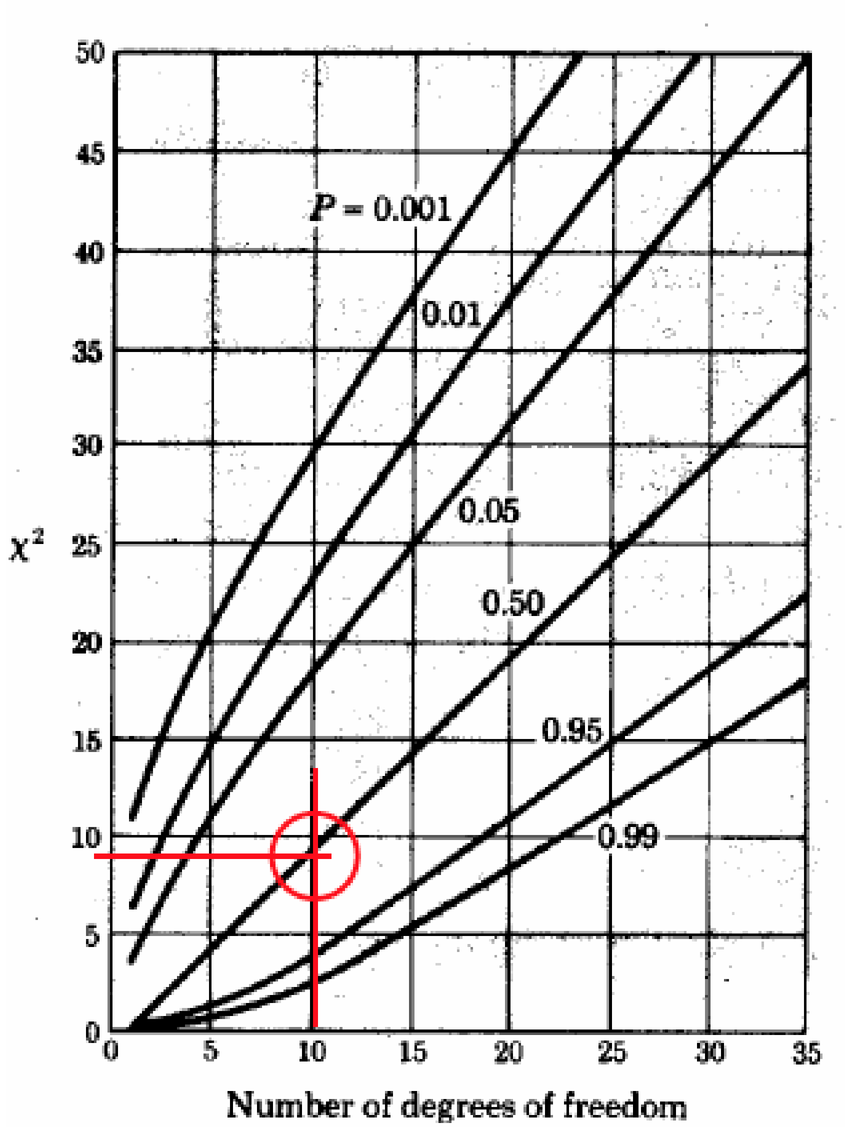
$$\chi^2 = \sum_{i=1}^{n} \frac{(measured_i - expected_i)^2}{expected_i}$$

In [76]:
```
gauss = 1./np.sqrt(2*np.pi*sigma**2)*exp(-((bin_edges[:-1]+dp/2.-mu)**2/(2*sigma**2)))
chisq = sum((hist - gauss)**2/gauss)
print '$\chi^2$ = %f' % chisq
```
```
$\chi^2$ = 6.126602
```

In [79]:
```
# degrees of freedom = number of bins minus the (order of the fit + 1):
print 'Number of degrees of freedom,  K - (m+1) = %d' % (K - 2)
```
```
Number of degrees of freedom,  K - (m+1) = 7
```

In [75]:
```
from scipy import stats
pval = 1 - stats.chi2.cdf(chisq, K-2); print 'Confidence level is %3.1f percent' % (pv
Image('../Lectures/Lecture2/chi_square_graph.png')
```
```
Confidence level is 52.5 percent
```

Out [75]:



we conclude that for the given set of measurements we are only 50% certain that we can use the Gaussian distribution assumptions

1.2 Calibration

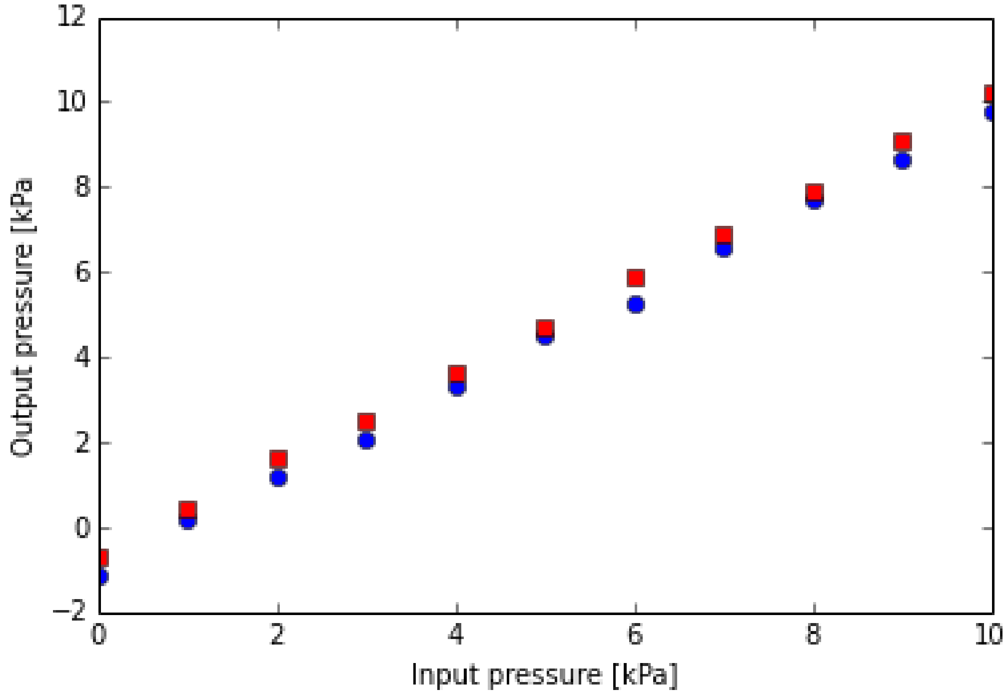
In [61]:
```
# Increasing pressure:
p_in_up = np.linspace(0.0,10.0,11)
p_out_up = np.array([-1.12, 0.21, 1.18, 2.09, 3.33, 4.50, 5.26, 6.59, 7.73, 8.68, 9.8]
```

In [62]:
```
# Decreasing pressure
p_in_down = np.flipud(p_in_up)
p_out_down = np.array([10.20, 9.10, 7.92, 6.89, 5.87, 4.71, 3.62, 2.48, 1.65, 0.42, -0
```

In [63]:
```
plot(p_in_up,p_out_up,'bo',p_in_down,p_out_down,'rs')
xlabel('Input pressure [kPa]')
ylabel('Output pressure [kPa]')
```

```
<matplotlib.text.Text at 0x107f39790>
```

Out [63]:



In [64]:
```
x = r_[p_in_up,p_in_down]
y = r_[p_out_up,p_out_down]
```

In [65]:
```
 polyfit(x,y,1)
```

Out [65]:
```
array([ 1.08231818, -0.84704545])
```

1.3 Estimate uncertainty

$q_0 = mq_i + b$

$q_0 = 1.08q_i + 0.85$

$\sigma_{q_0}^2 = \frac{1}{N}\sum(mq_i + b - q_0)$

We then use the inverse of the calibration curve to get the inputs from the outputs:

$q_i = \frac{q_0 - b}{m}$

$\sigma_{q_i}^2 = \frac{1}{N}\sum\left(\frac{q_0-b}{m} - q_i\right)^2 = \frac{\sigma_{q_0}^2}{m^2}$

In [66]:
```
m = 1.08
b = -0.85
std_q0 = sqrt(1./(y.size-1) * sum((m*x + b - y)**2))
print 'std(q_0) = %f ' % std_q0
```

```
std(q_0) = 0.203727
```

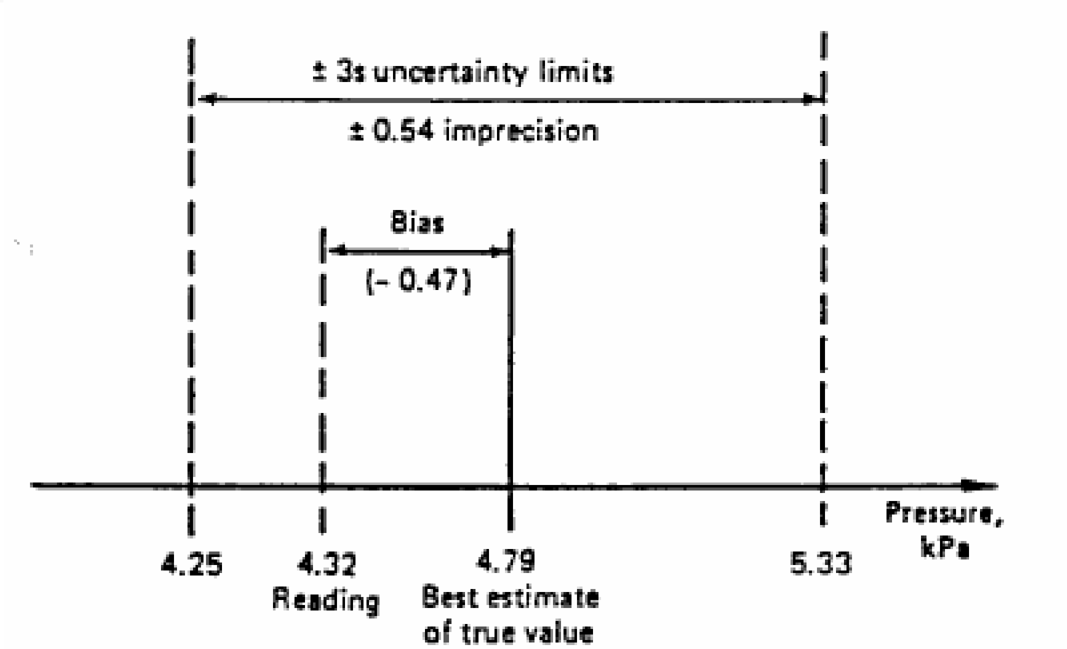
In [67]:
```
# let's assume we measured output
q_0 = 4.32 #kPa
# we estimate the real input as:
q_i = (q_0 - b)/m

# and its std. dev.

std_qi = std_q0/m
```

```
print 'q_i = %3.2f +- %3.2f kPa ' % (q_i, 3*std_qi)
```

q_i = 4.79 +- 0.57 kPa

In [68]:
```
# we can visualize the result as:
im = Image('../Lectures/Lecture2/result_pressure_measurement.png'); im
```

Out [68]:



Uncertainties of least-square best fit estimates:

$S_y^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - \overline{y})^2$

$S_{yx}^2 = \frac{1}{\nu}\sum_{i=1}^{N}(y_i - \overline{y_{c_i}})^2$

$\nu = N - (m+1)$

$S_m = S_{yx}^2 \frac{N}{N\sum_{i=1}^{N} x_i^2 - \left(\sum_{i=1}^{N} x_i\right)^2}$

$S_b = S_{yx}^2 \frac{N\sum_{i=1}^{N} x_i^2}{N\left[N\sum_{i=1}^{N} x_i^2 - \left(\sum_{i=1}^{N} x_i\right)^2\right]} S_m = 0.0134$ - sensitivity uncertainty

$S_b = 0.078$ - zero shift uncertainty

$m = 1.08 \pm 0.04$

$b = -0.85 \pm 0.24$ kPa