stats.sample Estimation of sample mean and variance

Estimation and sample statistics

The mean and variance definitions of Lec. prob.E and Lec. prob.moments apply only to a random variable for which we have a theoretical probability distribution. Typically, it is not until after having performed many measurements of a random variable that we can assign a good distribution model. Until then, measurements can help us *estimate* aspects of the data. We usually start by estimating basic parameters such as *mean* and *variance* before estimating a probability distribution.

There are two key aspects to randomness in the measurement of a random variable. First, of course, there is the underlying randomness with its probability distribution, mean, standard deviation, etc., which we call the *population statistics*. Second, there is the *statistical variability* that is due to the fact that we are *estimating* the random variable's statistics—called its *sample statistics*—from some sample. Statistical variability is decreased with greater sample size and number of samples, whereas the underlying randomness of the random variable does not decrease. Instead, our estimates of its probability distribution and statistics improve.

Sample mean, variance, and standard deviation

The *arithmetic mean* or **sample mean** of a measurand with sample size N, represented by random variable X, is defined as

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i. \tag{1}$$

If the sample size is large, $\overline{x} \to m_X$ (the sample mean approaches the mean). The **population**

mean is another name for the mean m_X, which is equal to

$$m_X = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} x_i. \tag{2}$$

Recall that the *definition* of the mean is $m_X = \mathrm{E}\left[x\right]$.

The **sample variance** of a measurand represented by random variable X is defined as

$$S_X^2 = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \overline{x})^2. \tag{3}$$

If the sample size is large, $S_X^2 \to \sigma_X^2$ (the sample variance approaches the variance). The **population variance** is another term for the variance σ_X^2, and can be expressed as

$$\sigma_X^2 = \lim_{N\to\infty} \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \overline{x})^2. \tag{4}$$

Recall that the *definition* of the variance is $\sigma_X^2 = \mathrm{E}\left[(X - m_X)^2\right]$.

The *sample standard deviation* of a measurand represented by random variable X is defined as

$$S_X = \sqrt{S_X^2}. \tag{5}$$

If the sample size is large, $S_X \to \sigma_X$ (the sample standard deviation approaches the standard deviation). The *population standard deviation* is another term for the standard deviation σ_X, and can be expressed as

$$\sigma_X = \lim_{N\to\infty} \sqrt{S_X^2}. \tag{6}$$

Recall that the *definition* of the standard deviation is $\sigma_X = \sqrt{\sigma_X^2}$.

Sample statistics as random variables

There is an ambiguity in our usage of the term “sample.” It can mean just one measurement or it can mean a collection of measurements gathered together. Hopefully, it is clear from context.

In the latter sense, often we collect multiple samples, each of which has its own sample mean $\overline{X}_i$ and standard deviation S_{X_i}. In this situation, $\overline{X}_i$ and S_{X_i} are themselves random variables (meta af, I know). This means they have their own sample means $\overline{\overline{X}_i}$ and $\overline{S_{X_i}}$ and standard deviations $S_{\overline{X}_i}$ and $S_{S_{X_i}}$.

The **mean of means** $\overline{\overline{X}_i}$ is equivalent to a mean with a larger sample size and is therefore our best estimate of the mean of the underlying random process. The **mean of standard deviations** $\overline{S_{X_i}}$ is our best estimate of the standard deviation of the underlying random process. The **standard deviation of means** $S_{\overline{X}_i}$ is a measure of the spread in our estimates of the mean. It is our best estimate of the standard deviation of the statistical variation and should therefore tend to zero as sample size and number of samples increases. The **standard deviation of standard deviations** $S_{S_{X_i}}$ is a measure of the spread in our estimates of the standard deviation of the underlying process. It should also tend to zero as sample size and number of samples increases.

Let N be the size of each sample. It can be shown that the standard deviation of the means $S_{\overline{X}_i}$ can be estimated from a single sample standard deviation:

$$S_{\overline{X}_i} \approx \frac{S_{X_i}}{\sqrt{N}}. \tag{7}$$

This shows that as the sample size N increases, the statistical variability of the mean decreases (and in the limit approaches zero).

Nonstationary signal statistics

The sample mean, variance, and standard deviation definitions, above, assume the random process is *stationary*—that is, its population mean does not vary with time. However, a great many measurement signals have populations that *do* vary with time, i.e. they are *nonstationary*. Sometimes the nonstationarity arises from a "drift" in the dc value of a signal or some other slowly changing variable. But dynamic signals can also change in a recognizable and predictable manner, as when, say, the temperature of a room changes when a window is opened or when a water level changes with the tide.

Typically, we would like to minimize the effect of nonstationarity on the signal statistics. In certain cases, such as drift, the variation is a nuissance only, but other times it is the point of the measurement.

Two common techniques are used, depending on the overall type of nonstationarity. If it is periodic with some known or estimated period, the measurement data series can be "folded" or "reshaped" such that the ith measurement of each period corresponds to the ith measurement of all other periods. In this case, somewhat counterintuitively, we can consider the ith measurements to correspond to a sample of size N, where N is the number of periods over which measurements are made.

When the signal is aperiodic, we often simply divide it into "small" (relative to its overall trend) intervals over which statistics are computed, separately.

Note that in this discussion, we have assumed that the nonstationarity of the signal is due to a variable that is deterministic (not random).

Example stats.sample-1 re: measurement of gaussian noise on nonstationary signal

Consider the measurement of the temperature inside a desktop computer chassis via an inexpensive *thermistor*, a resistor that changes resistance with temperature. The processor and power supply heat the chassis in a manner that depends on processing demand. For the test protocol, the processors are cycled sinusoidally through processing power levels at a frequency of 50 mHz for $n_T = 12$ periods and sampled at 1 Hz. Assume a temperature fluctuation between about 20 and 50 C and gaussian noise with standard deviation 4 C. Consider a *sample* to be the multiple measurements of a certain instant in the period.

1. Generate and plot simulated temperature data as a time series and as a histogram or frequency distribution. Comment on why the frequency distribution sucks.
2. Compute the sample mean and standard deviation *for each sample in the cycle*.
3. Subtract the mean from each sample in the period such that each sample distribution is centered at zero. Plot the composite frequency distribution of all samples, together. This represents our best estimate of the frequency distribution of the underlying process.
4. Plot a comparison of the theoretical mean, which is 35, and the sample mean of means with an error bar. Vary the number of samples n_T and comment on its effect on the estimate.
5. Plot a comparison of the theoretical standard deviation and the sample mean of sample standard deviations with an error bar. Vary the number of samples n_T and comment on its effect on the estimate.
6. Plot the sample means over a single period with error bars of ± one sample standard deviation of the means. This represents our best estimate of the

sinusoidal heating temperature. Vary the number of samples n_T and comment on the estimate.

```
clear; close all; % clear kernel
```

Generate the temperature data

The temperature data can be generated by constructing an array that is passed to a sinusoid, then “randomized” by gaussian random numbers. Note that we add `1` to `np` and `n` to avoid the sneaky fencepost error.

```
f = 50e-3; % Hz ... sinusoid frequency
a = 15; % C ... amplitude of oscillation
dc = 35; % C ... dc offset of oscillation
fs = 1; % Hz ... sampling frequency
nT = 12; % number of sinusoid periods
s = 4; % C ... standard deviation
np = fs/f+1; % number of samples per period
n = nT*np+1; % total number of samples

t_a = linspace(0,nT/f,n); % time array
sin_a = dc + a*sin(2*pi*f*t_a); % sinusoidal array
rng(43); % seed the random number generator
noise_a = s*randn(size(t_a)); % gaussian noise
signal_a = sin_a + noise_a; % sinusoid + noise
```

Now that we have an array of “data,” we’re ready to plot.

```
h = figure;
p = plot(t_a,signal_a,'o-',...
    'Color',[.8,.8,.8],...
    'MarkerFaceColor','b',...
    'MarkerEdgeColor','none',...
    'MarkerSize',3);
xlabel('time (s)');
ylabel('temperature (C)');
hgsave(h,'figures/temp');
```

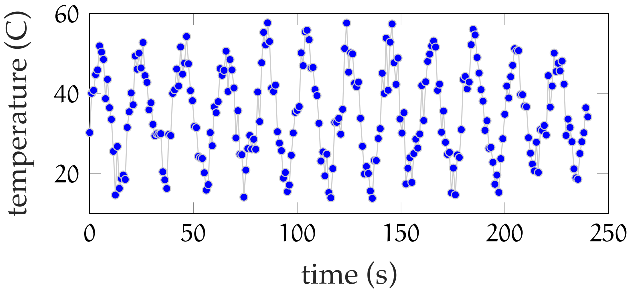


Figure sample.1: temperature over time

This is something like what we might see for continuous measurement data. Now, the histogram.

```
h = figure;
histogram(signal_a,...
    30, ... % number of bins
    'normalization','probability'... % for PMF
);
xlabel('temperature (C)')
ylabel('probability')
hgsave(h,'figures/temp');
```

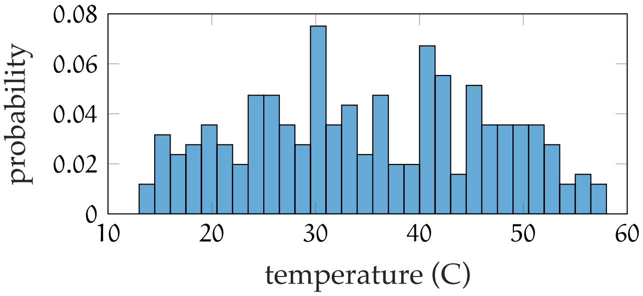


Figure sample.2: a poor histogram due to unstationarity of the signal.

This sucks because we plot a frequency distribution to tell us about the random variation, but this data includes the sinusoid.

Sample mean, variance, and standard deviation

To compute the sample mean μ and standard deviation s for each sample in the period, we must "pick out" the `nT` data points that correspond to each other. Currently, they're in one long 1 × n array signal_a. It is helpful to *reshape* the data so it is in an nT × np array, which each row corresponding to a new period. This leaves the correct points aligned in columns. It is important to note that we can do this "folding" operation only when we know rather precisely the period of the underlying sinusoid. It is given in the problem that it is a controlled experiment variable. If we did not know it, we would have to estimate it, too, from the data.

```
signal_ar = reshape(signal_a(1:end-1)',[np,nT])';
size(signal_ar) % check size
signal_ar(1:3,1:4) % print three rows, four columns
```

```
ans =

    12    21


ans =

   30.2718   40.0946   40.8341   44.7662
   40.1836   37.2245   49.4076   46.1137
   40.0571   40.9718   46.1627   41.9145
```

Define the mean, variance, and standard deviation functions as "anonmymous functions." We "roll our own." These are *not* as efficient or flexible as the built-in *Matlab* functions mean, var, and std, which should typically be used.

```
my_mean = @(vec) sum(vec)/length(vec);
my_var = @(vec) sum((vec-my_mean(vec)).^2)/...
  (length(vec)-1);
my_std = @(vec) sqrt(my_var(vec));
```

Now the sample mean, variance, and standard deviations can be computed. We proceed by looping through each column of the reshaped signal array.

```
mu_a = NaN*ones(1,np); % initialize mean array
var_a = NaN*ones(1,np); % initialize var array
```

```
s_a = NaN*ones(1,np); % initialize std array

for i = 1:np % for each column
    mu_a(i) = my_mean(signal_ar(:,i));
    var_a(i) = my_var(signal_ar(:,i));
    s_a(i) = sqrt(var_a(i)); % touch of speed
end
```

Composite frequency distribution

The columns represent samples. We want to subtract the mean from each column. We use `repmat` to reproduce `mu_a` in `nT` rows so it can be easily subtracted.

```
signal_arz = signal_ar - repmat(mu_a,[nT,1]);
size(signal_arz) % check size
signal_arz(1:3,1:4) % print three rows, four columns
```

```
ans =

    12    21


ans =

   -5.0881    0.9525   -0.2909   -1.5700
    4.8237   -1.9176    8.2826   -0.2225
    4.6972    1.8297    5.0377   -4.4216
```

Now that all samples have the same mean, we can lump them into one big bin for the frequency distribution. There are some nice built-in functions to do a quick reshape and fit.

```
% resize
signal_arzr = reshape(signal_arz,[1,nT*np]);
size(signal_arzr) % check size
% fit
pdfit_model = fitdist(signal_arzr','normal'); % fit
x_a = linspace(-15,15,100);
pdfit_a = pdf(pdfit_model,x_a);
pdf_a = normpdf(x_a,0,s); % theoretical pdf
```

```
ans =

     1   252
```

Plot!

```
h = figure;
histogram(signal_arzr,...
    round(s*sqrt(nT)), ... % number of bins
    'normalization','probability'... % for PMF
);
hold on
plot(x_a,pdfit_a,'b-','linewidth',2); hold on
plot(x_a,pdf_a,'g--','linewidth',2);
legend('pmf','pdf est.','pdf')
xlabel('zero-mean temperature (C)')
ylabel('probability mass/density')
hgsave(h,'figures/temp');
```

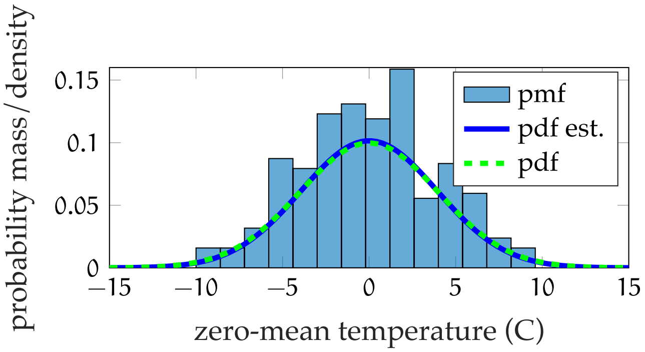


Figure sample.3: PMF and estimated and theoretical PDFs.

Means comparison

The sample mean of means is simply the following.

```
mu_mu = my_mean(mu_a)
```

```
mu_mu =

   35.1175
```

The standard deviation that works as an error bar, which should reflect how well we can estimate the point plotted, is the standard deviation of the means. It is difficult to compute this directly for a nonstationary process. We use the estimate given above and improve upon it by using the mean of standard deviations instead of a single sample's standard deviation.

```
s_mu = mean(s_a)/sqrt(nT)
```

```
s_mu =

    1.1580
```

Now, for the simple plot.

```
h = figure;
bar(mu_mu); hold on % gives bar
errorbar(mu_mu,s_mu,'r','linewidth',2) % error bar
ax = gca; % current axis
ax.XTickLabels = {'$\overline{\overline{X}}$'};
ax.TickLabelInterpreter = 'latex';
hgsave(h,'figures/temp');
```

Standard deviations comparison

The sample mean of standard deviations is simply the following.

```
mu_s = my_mean(s_a)
```

```
mu_s =

    4.0114
```

The standard deviation that works as an error bar, which should reflect how well we can estimate the point plotted, is the standard deviation of the standard deviations. We can compute this directly.

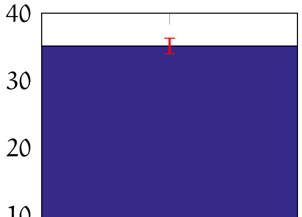
```
s_s = my_std(s_a)
```

```
s_s =

    0.8495
```

Now, for the simple plot.

```
h = figure;
bar(mu_s); hold on % gives bar
errorbar(mu_s,s_s,'r','linewidth',2) % error bars
ax = gca; % current axis
ax.XTickLabels = {'$\overline{S_X}$'};
ax.TickLabelInterpreter = 'latex';
hgsave(h,'figures/temp');
```



Plot a period with error bars

Plotting the data with error bars is fairly straightforward with the built-in `errorbar` function. The main question is “which standard deviation?” Since we’re plotting the means, it makes sense to plot the error bars as a single sample standard deviation of the means.

```
h = figure;
e1 = errorbar(t_a(1:np),mu_a,s_mu*ones(1,np),'b');
hold on
t_a2 = linspace(0,1/f,101);
e2 = plot(t_a2,dc + a*sin(2*pi*f*t_a2),'r-');
xlim([t_a(1),t_a(np)])
grid on
xlabel('folded time (s)')
ylabel('temperature (C)')
legend([e1 e2],'sample mean','population mean',...
'Location','NorthEast')
hgsave(h,'figures/temp');
```

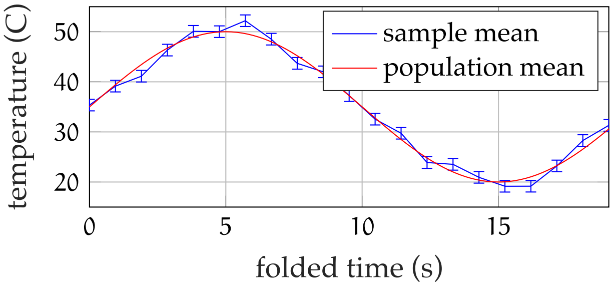


Figure sample.6: sample means over a period.