# Linear regression exercise

In [ ]:
```
using Gadfly, Distributions
```

We first define the polynomial basis function: $$\phi_\text{poly} = [1, x, x^2, \ldots, x^d]\\$$

In [ ]:
```
phi_poly(x,d) = x.^((0:d)')                    # polynomial basis function
```

The maximum likelihood estimator can be written as $$w_\text{MLE} = (X'*X)^{-1}X'y$$

In [ ]:
```
wMLE(xtrain, ytrain, phi) = begin
    Xtrain = phi(xtrain)
    wpred  = (Xtrain' * Xtrain) \ (Xtrain' * ytrain)
end
```

Finally, we define a function for the mean squared error.

In [ ]:
```
mse(yest, ytrue) = mean((yest-ytrue).^2)
```

For convenience we pack everything into some clever .
Note that the arguments `wpred` and `phi` are functions.

In [ ]:
```
lrplot(x, y, w, phi, titleprefix="") = begin
    n      = 100                         # number of test points
    xline  = linspace(minimum(x), maximum(x), 100)
    yline  = phi(xline)*w
    yerr   = mse(phi(x)*w, y)
    plot(layer(x=x,      y=y,     Geom.point),
         layer(x=xline , y=yline, Geom.line),
    Guide.title("$titleprefix error $yerr"))
end
```

# Example with toy data

In [ ]:
```
readtt(fname) = begin
    x = readdlm(fname, ',')
    (x[:,1], x[:,2])
end
xtrain, ytrain = readtt("TrainingSet1D.csv")
xtest , ytest  = readtt("TestSet1D.csv")
```

## Fit via MLE and compare training and test error

In [ ]:
```
phi(x) = phi_poly(x, 1)        # define a basis function
w      = wMLE(xtrain, ytrain, phi)
set_default_plot_size(25cm, 10cm)
hstack(lrplot(xtrain, ytrain, w, phi, "training"), lrplot(xtest, ytest, w, phi, "test"))
```