(*The Coleman algorithm*). The following question requires you to write MATLAB (*not* pseudo) code. Assume a momentary objective function of the form

$$U(k, k') = \frac{[k^\alpha + (1-\delta)k - k']^{1-\rho}}{1-\rho}, \text{for } \rho \geq 0.$$

Write out the computer code using MATLAB syntax for solving the dynamic programming problem

$$V(k) = \max_{k'} \{U(k, k') + \beta V(k')\}.$$

In the above dynamic programming problem (to be solved on the computer), the choice variable, k', is to formulated as *continuous* variable, not a discrete one. This is to be done by solving the non-linear first-order condition associated with the above dynamic programming problem. The value function $V(k)$ is to be approximated by a quadratic function on the grid $\mathfrak{K} = \{k_1, \cdots, k_n\}$; i.e., the function V is fit using a *quadratic* function on the domain $[k_1, k_n]$. Let $\alpha = 0.3$, $\delta = 0.1$, $\beta = 0.96$, and $\rho = 2$. Assume that the grid spans 10,001 points centered on the steady-state value for capital, k^*, spanning the interval $[k^* - .5k^*, k^* + .5k^*]$. Your algorithm *must* contain the following steps somewhere:

1. An m file that fits a quadratic function to an *arbitrary* function V, given values for the function V on the grid $\mathfrak{K}$. Information on the POLYFIT function in MATLAB is provided below.

2. Given the function $U(k, k')$ and a guess for a quadratic function for V, compute a solution for k' at each grid point for k in $\mathfrak{K}$ *using the non-linear first-order condition* associated with the above dynamic programming problem. Note that the solution for k' will *not* in general be a point in the grid $\mathfrak{K}$. This is to be done using the FZERO function in MATLAB. Construct a revised guess for V at each grid point $k \in \mathfrak{K}$.

The following information in MATLAB may help:

POLYFIT Fit polynomial to data.

P = polyfit(X,Y,N) finds the coefficients of a polynomial P(X) of degree N that fits the data Y best in a least-squares sense. P is a row vector of length N+1 containing the polynomial coefficients in descending powers, P(1)*X^N + P(2)*X^(N-1) +...+ P(N)*X + P(N+1).

FZERO Single-variable nonlinear zero finding.

X = fzero(FUN,X0) tries to find a zero of the function FUN near X0, if X0 is a scalar.