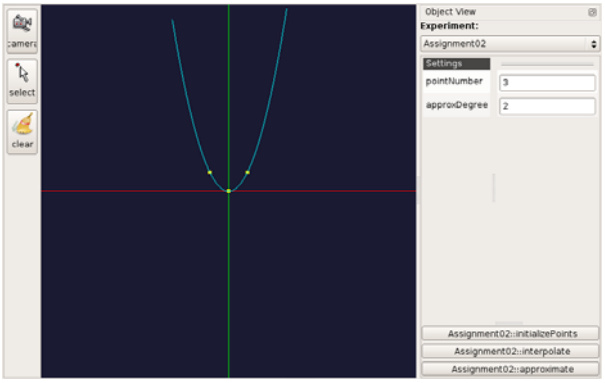
Dr. Michael Wand

Geometric Modeling

Assignment sheet 2 (Interpolation and Approximation, due May 14th 2010)

---

(1) Polynomial Interpolation [2+8+2 points]



a. Create a new experiment with a GLGeometryViewer for 2D visualization. Add a button that inserts *n* distinct points to the viewport, which all lie on the parabola $y=x^2$. Allow the user to set the value of *n* interactively. A reasonable default will be *n=3*.

b. Add a button which interpolates the *n* points using a polynomial of suitable degree. Plot the polynomial in the viewport. Choose the range of the plot sufficiently large, so that you can observe the behavior of the polynomial in front of the first and behind the last point. Try various settings of *n* (e.g., *n=7*) and move around the points. How does the resulting polynomial change?
Hints: Helpful math routines can be found in DynamicLinearAlgebra.h. You may approximate the polynomial by short lines to plot it.

c. In (b), you needed to invert a matrix. In which case will this fail? Make sure your program reports a useful error message to the user.
Hint: In GeoX, invertMatrix() throws an exception when it fails. Your C++ code to treat it should look like this: try { … invertMatrix(…); … } catch (PException& e) { output << "Tell the user what went wrong and how to avoid it.\n"; }

(2) Polynomial Approximation [8 points]

a. In the experiment from (1), add a button which approximates the *n* points using a polynomial of user-defined degree *m* in a least squares sense. A reasonable default will be *m=2*. You do not need to treat the case in which *m* is greater than the degree you would use for interpolation (just produce a useful error message). Try various settings of *n* and *m* and move around the points. Compare your results to an interpolation of the points.