The Big Picture

- Given an arbitrary function $f$ and an interval $[a, b]$, I want to have an automatic procedure that will draw the graph of the function over the interval.
- Using the procedure should be as easy as this:
  
  ```
  plot(f, a, b)
  ```
- At this level of generality the problem is “hard” but, for more restricted classes of function, progress can be made.
- This talk will mostly consider smooth (i.e. differentiable) functions but will also mention piecewise-smooth functions.

Curves and Polylines

- In computer graphics, a polyline is a connected set of line segments.
- A standard method for drawing parametric curves in a vector drawing system uses a polyline approximation based on (many) equally-spaced parameter values.
- Drawing the graph of a function $f(x)$ over an interval $[a, b]$ is an important special case.
- This is the method used in the R function `curve`, which uses 101 equally-spaced points by default.

What Makes a Polyline Approximation Good?

- Even when the accuracy of the polyline approximation is good, it may not look “right” because it is clearly not a smooth curve.
- This is because the visual system detects the “corners” at the joins of the segments making up the polyline.
- This is very much like the edge-detection process built into the visual system.
- The real issue here is one of visual perception rather than mathematical accuracy.
Visual Detection of Polyline Joins

- As with many perceptual phenomena, there appears to be a threshold effect for detection of a join between polylines.
- When the change direction, $\theta$, of the polylines is "small" (e.g. $< 3^\circ$), no join is detected and the transition from one segment to the next is seen as "smooth."

Drawing Visually Smooth Polylines

- To draw a polyline approximation to a curve that is visually smooth:
  - Choose points along the curve so that the change of angle from segment to segment is less than the "visually smooth" threshold.
  - Draw the polyline that joins the points.

Mathematical “Formulation”

- Given a smooth function $f : [a, b] \to \mathbb{R}$, let $\phi(x)$ be the angle that the tangent to $f(x)$ at $x$ makes with the $x$-axis.
- We want to choose $a = x_0 < x_1 < x_2 < \cdots < x_n = b$ so that the change of angle from $x_i$ to $x_{i+1}$ is small.
- I.e., for some suitably small $\delta$ the $x_i$ should satisfy
  $$|\phi(x_{i+1}) - \phi(x_i)| < \delta$$
  for $i = 1, \ldots, n$.
- How can the $x_i$ values be chosen to make this happen?

A Solution

The function

$$h(x) = \int_a^x |\phi'(t)| \, dt$$

is a monotonically non-decreasing function of $x$ and so has a well-defined inverse. Further, if $a < v < w$,

$$|\phi(v) - \phi(u)| \leq \int_u^v |\phi'(t)| \, dt = h(v) - h(u)$$

Choose points $h(a) = y_0 < y_1 < y_2 < \cdots < y_n = h(b)$ such that $y_{i+1} - y_i < \delta$ for all $i$. Set $x_i = h^{-1}(y_i)$ then the points

$$(x_0, f(x_0)), (x_1, f(x_2)), \ldots, (x_n, f(x_n))$$

define the approximating polyline.

Choice of Plotting Ordinates

Computer Implementation

- To implement the method in a computer the continuous problem is approximated by its discrete version using a very finely spaced grid of points splitting up the interval from $a$ to $b$.
  - Derivatives are replaced by differences.
  - Indefinite integrals are replaced by cumulative sums.
- Linear interpolation can be used to compute $h(x)$ and $h^{-1}(u)$.

An R Implementation

```r
smoothx = function(f, a, b, delta = 2.5, n = 1001) {
  dr = delta * pi / 180
  x = seq(a, b, length = n)
  y = f(x)
  phi = atan2(ay * diff(y), ax * diff(x))
  d.phi = c(0, diff(phi), 0)
  u = csum(abso(d.phi) + 1e-3/n)
  if (u[n] - u[1] < dr)
    x[c(1, length(x))]
  else
    approx(u, x, n = ceiling((u[n] - u[1])/dr) + 1)$y
}
```
There are many (useful/interesting) functions that are not continuous on their domain or which lie partly outside the plotting window of interest.

Extending the drawing method outlined previously requires decomposing the domain of a function into intervals where the smooth polyline approximation can be used.

This means dealing with:
- Regions where the function is out of the viewing window (clipping).
- Points where the function is not defined.
- Simple function discontinuities.
- Asymptotes.

These (software) patents make work in this area problematic.
Summary

• A novel method of curve drawing based on visual smoothness has been presented.

• The method can be incorporated into a general system for automatically drawing piecewise smooth functions.

• United States software patents make the delivery of this technology problematic. (Oppose the TPP!)