Differentiating Through
Conjugate gradient.

Bruce Chrisman

University of Hartford, Hartford.
Conjugate Gradient

Solve \( Ax = b \)

then solve \( A\tilde{x} = b - \tilde{A}x \)

Alternatively,

just apply CG to \( Ax = b \)

but re-declare everything to be

a (truncated) Taylor series

\[
A = A^{(0)} + A^{(1)}t + A^{(2)}t^2 + \ldots
\]

\[
b = b^{(0)} + b^{(1)}t + b^{(2)}t^2 + \ldots
\]

\[
x = x^{(0)} + x^{(1)}t + x^{(2)}t^2 + \ldots
\]

depending on how many derivatives we want.
the CG algorithm:

start: set $i := 0$

$g_0 = Ax_0 - b$
$p_i = -g_0$

loop: $i := i + 1$

choose $\alpha$: s.t. $\alpha_i (p_i^T A p_i) = g_i^2$

$x_i = x_{i-1} + \alpha_i p_i$

$s_i = Ax_i - b = g_{i-1} + \alpha_i A p_i$

if $\|g_i\| < \varepsilon$ then terminate

choose $\beta$: s.t. $\beta_i g_i^2 = s_i^2$

$p_{i+1} = \beta_i p_i - g_i$

go to loop

remember: all the variables are truncated Taylor series
Assert: \( V_j \neq i \)

\[ s_i \cdot g_j = 0 \text{ and } p_i^T A p_j = 0 \]

as Taylor series - i.e. for all powers of \( t \)

Proof by induction;

\( i = j + 1 \) holds by definition of \( a_{j+1} \) and \( p_j \)

Then for \( i > j + 1 \)

\[ p_i^T A p_j = (p_i - s_j)^T A p_j = -s_j^T A p_j \quad \text{by induction on } p_i \]

so

\[ d_j p_i^T A p_j = -d_j s_i^T A p_j = -s_i^T (g_j - g_{j+1}) = 0 \]

by induction of \( g_j \)

[Note: This requires \( d_j^{(0)} \neq 0 \); we'll come back to this!]

and

\[ s_{i+1} \cdot g_j = (s_c + d_{i+1} p_{i+1})^T g_j \quad (s_i \cdot g_j = 0) \]

\[ = d_{i+1} p_{i+1}^T A (p_j - p_{j+1}) = 0 \]
So all goes well until we have \( g_i^{(0)} = 0 \) at which point \( A^{(0)} x_i^{(0)} = b^{(0)} \).

If we also have \( g_i^{(1)} = 0 \) then since

\[
g_i = A x_i - b
\]

we have

\[
g_i^{(1)} = A^{(1)} x_i^{(1)} + A^{(0)} x_i^{(0)} - b^{(1)} = 0
\]

so \( x_i^{(1)} \) is the solution to

\[
A^{(0)} x_i^{(1)} = b^{(1)} - A^{(0)} x_i^{(0)}
\]

as required.

But what if at \( i = i_0 \) we have

\[
g_i^{(0)} = 0 \quad g_i^{(1)} \neq 0 ?
\]

Answer: just keep going!
\[ \beta_{i_0} = \frac{g_{i_0}^2}{g_{i_0-1}^2} \]

so \( \beta_{i_0}^{(0)} = \beta_{i_0}^{(1)} = 0 \).

Therefore:

\[ p_{i_0+1}^{(0)} = 0 \quad p_{i_0+1}^{(1)} = -g_{i_0}^{(1)} \]

\[ \alpha_{i_0+1}^{(0)} = \frac{(g_{i_0}^{(1)})^2}{p_{i_0+1}^{(1)} \tau A_{i_0}^{(0)} p_{i_0+1}^{(1)}} \neq 0 \]

(using d'Hospital's rule, twice)

and on we go, with \( g_{i_0}^{(0)} = p_{i_0}^{(0)} = 0 \).

Note that we need truncated Taylor series of order at least two even for first derivatives.
At first sight, we seem to just have a CG restart at \( g_i \) ... but

**Assert:** taking the sequence

\[ g_i^{(0)} \text{ for } i \leq i_0 \text{ then } g_i^{(n)} \text{ for } i > i_0 \]

gives an orthogonal sequence

**Proof:** for \( i > j \) we have \( g_i \cdot g_j = 0 \), so

\[ g_i^{(0)} g_j + g_i^{(n)} g_j = 0 \]

so for \( i > i_0 > j \) we have \( g_i^{(0)} = 0 \) giving

\[ g_i \perp g_j^{(n)} \]

also

\[ g_i^{(0)} g_j + g_i^{(1)} g_j + g_i^{(2)} g_j = 0 \]

so for \( i > j > i_0 \) we have \( g_i^{(0)} = g_j^{(0)} = 0 \) giving

\[ g_i \perp g_j \]
There is only a limited number of linearly independent directions, so eventually we have $g_i$ with $g_i^{(n)} = g_i^{(n')} = 0$.

But - no need to stop there.

Can now switch to $g_i^{(2)}$ and continue to get $g_i^{(n)}$, and so on, all in at most $n$ steps!
Interfacing to the AD tool:

The tool needs to know when a term is small enough to be zero:

\[ \frac{a + 6t}{c + dt} \text{ equal to } \frac{9}{c} \text{ or to } \frac{6}{d} ? \]

The programmer already has a view about this but she needs a way to tell the AD tool.
When singularities are involved, we need a way to feel the AD tool that the primal has converged but the tangent process hasn't.

Again, the programmer knows...
perhaps I want to
restart the primal
but continue the tangent process