Computing the hypergeometric function 2F1 using a recipe of Gosper.

AVt, Apr 2007

Bill Gosper was cited by Richard Fateman with the following text, how one can compute the Gauss function 2F1:

```
...
Here is how Macsyma (and Mathematica, at least as of a few years ago)
handle this.  Rejoice:  we don't need to get lucky--we can have our
z^2/(4 z - 4) convergence unconditionally if we switch from a series
to a first order recurrence on three variables.  Specifically, let
```

$$[d_0, e_0, f_0] = [0, 1, 0],$$

$$\begin{bmatrix} d_{k+1} \\ e_{k+1} \\ f_{k+1} \end{bmatrix} = \begin{bmatrix} \dfrac{(k+c-b-a)\,d_k\,z}{k} \\ (k+a)(k+b)\,z\left(e_k - \dfrac{d_k}{1-z}\right) \\ \dfrac{(k+1)(k+\frac{c}{2})(k+\frac{c+1}{2})}{(k+1)(k+\frac{c}{2})(k+\frac{c+1}{2})} \\ \dfrac{a\,b\,d_k\,z}{k} \\ (k+a)(k+b)\,z\left(\dfrac{d_k}{1-z} + (k+c)\,e_k\right) \\ \dfrac{(k+1)(k+\frac{c}{2})(k+\frac{c+1}{2})}{(k+1)(k+\frac{c}{2})(k+\frac{c+1}{2})} \\ \dfrac{d_k\,(k((c-b-a)z + k(z-2) - c) - a\,b\,z)}{k} \\ f_k - \dfrac{d_k\,(k((c-b-a)z + k(z-2) - c) - a\,b\,z)}{k\,(k+\frac{c}{2})(1-z)} + e_k \end{bmatrix}$$

```
Then d_k and e_k approach 0 like (z^2/(4z-4))^k, and the running sum f_k
approaches 2F1(a,b;c|z).
...
```

In a private mail Bill Gosper added, that the recursion holds for those z which are inside (where "inside" means winding number = 1) `the prolate cardioid parametrized by [sqrt(8)*cos(t)-2*cos(2*t), sqrt(8)*sin(t)-2*sin(2*t)].`

One can use this to write code, which evaluates 2F1 using double precision (with some limitations on exactness of course).
This is *much* simplier than the solution given by Forrey (his paper covers the real case, not sure about the complex case, since everything it is in Fortran which I do not speak).

The idea is, that with the above the linear transformations given in A&S need to be applied only <u>once</u> to cover the whole complex plane: up to 1 transformation one can either use Gosper's recursion or even lives in the unit circle with radius 1/2 (where the hypergeometric series already converges quite fast).

Most troubles are given for exceptional parameters, where the transformations in A&S are not applicable and would involve expensive (and less exact ?) series in Digamma functions. However choosing some appropriate presentation (which can be found by reading the proof in Lebedev's book and massaging them in Maple) one can treat this by numerical differentiating w.r.t. parameters. And choosing enough points one can take a stepsize large enough to keep errors small.

PS: values in the branch cut (Reals greater than 1) are treated as in Maple (counter-clock wise w.r.t. the branching point), which means coming from the lower half plane. In the branch point z=1 a dummy value is returned (expect of the classical case).

<u>References</u>:
Gosper: posting at http://www.math.utexas.edu/pipermail/maxima/2006/000126.html (by Fateman) and private mails
Abramowitz & Stegun: The Handbook
Lebedev: Special Functions, Dover Publications
Forrey: Computing the Hypergeometric Function (1997), J of comp Physics

```
> restart; interface(version); Digits:=16: with(plots):
```

Classic Worksheet Interface, Maple 11.00, Windows, Feb 16 2007 Build ID 277223

Visualize the curve to see, what the transformations do with it:

```
> fx:= t -> sqrt(8)*cos(t)-2*cos(2*t);
  fy:= t -> sqrt(8)*sin(t)-2*sin(2*t);
```

$$fx := t \to \sqrt{8}\cos(t) - 2\cos(2t)$$

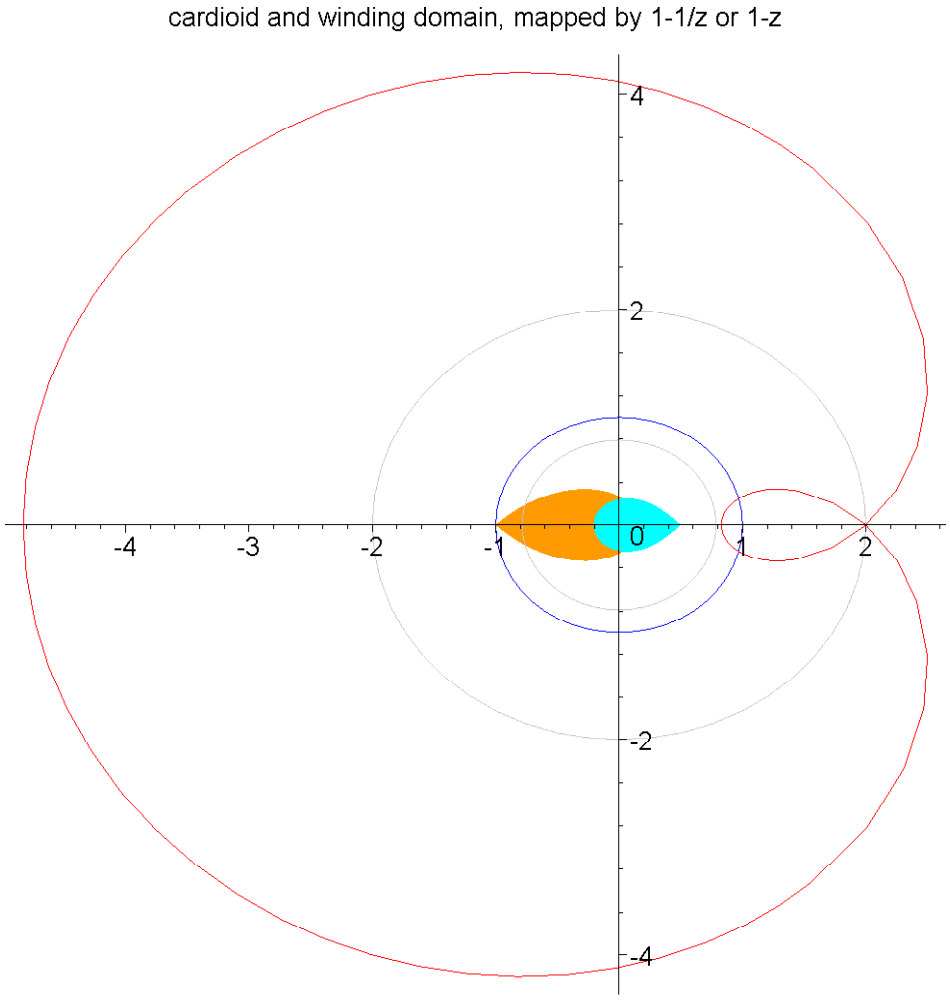
$$fy := t \to \sqrt{8}\sin(t) - 2\sin(2t)$$

```
> tau4:= z -> 1 - z;
  tau5:= z -> 1 - 1/z;

  'tau4( fx(t) + fy(t)*I )':
  [Re(%),Im(%)] assuming assumptionsW2:
  plot([op(%), t=-Pi/4..Pi/4], color=coral,filled=true, title="winding domain mapped by 1-z"):
  P4:=%:

  'tau5( fx(t) + fy(t)*I )':
  [Re(%),Im(%)] assuming assumptionsW2:
  plot([op(%), t=-Pi/4..Pi/4], color=cyan,filled=true, title="winding domain mapped by 1-1/z"):
  P5:=%:
```

$$\tau 4 := z \to 1 - z$$

$$\tau 5 := z \to 1 - \frac{1}{z}$$

```
> cart:=plot([fx(t),fy(t), t=-Pi..Pi],
    scaling=unconstrained, title= "cardioid and winding domain, mapped by 1-1/z or 1-z"):
  plot([cos(t), sin(t), t=-Pi..Pi], color=blue, scaling=unconstrained):
  plot([2*cos(t), 2*sin(t), t=-Pi..Pi], color=grey, scaling=unconstrained):
  plot([Pi/4*cos(t), Pi/4*sin(t), t=-Pi..Pi], color=grey, scaling=unconstrained):
  display(cart,%%%,%%,%, P5, P4);
  P_all:=%:
```



The red curve is the cardioid.

For the domain having winding number 2 (a region containing 1 and having 2 on its boundary) the recursion can not be used. But using 1 - 1/z or 1 - z (the colored domains in the picture) brings it into half the unit circle (just stare at it and play with abs(z) being larger or smaller than 3/2).

The domain with winding number 2 can be characterized by an algebraic curve (fiddling with Maple), so one has a handy criterion for coding in C (however keep away from the boundaries to have more stability).

```
> W2:= x -> sqrt(-(x^2 + 4*sqrt(5-2*x) - 8));
  plot(W2(x), x=  2*2^(1/2)-2 ... 2):
  plot(-W2(x), x=  2*2^(1/2)-2 ... 2, color=magenta, title="domain for winding number = 2"):
  display(%,%%);
  assumptionsW2:= 2*2^(1/2)-2<= x, x<=2, Pi/4<= t, t<=Pi/4 ;
```

$$W2 := x \to \sqrt{-x^2 - 4\sqrt{5 - 2x} + 8}$$

$$assumptionsW2 := 2\sqrt{2} - 2 \le x,\ x \le 2,\ \frac{\pi}{4} \le t,\ t \le \frac{\pi}{4}$$

For z being larger than 2 in size one transforms by 1/z of course. Hence with only 1 transformation one can achieve a convenient situation.

Almost.

There are the ugly cases of parameters, where the transforms can not be used and lead to the formulae A&S 15.3.10 and beyond.

However one can write all the transformed hypergeometric functions in the form of $\frac{\pi}{\sin(\pi\,\mu)}$ * $(\,g1(\mu) - g2(\mu)\,)$ (up to factors involving Gamma) where μ is the ugly parameter sum and g1, g2 are holomorphic and identical for $\mu = 0$.

So one can apply l'Hospital's rule. Done - purely numerical.

Note that double precision has limitations, say: already (t+1)/(t-1) = hypergeom([-1,-1],[t-1],2) will give serious errors for t close but not equal to 1, so one can not hope for highly exact results in any case and hypergeometric functions cover a wide class.

Another example is $\mathbf{hypergeom}([1, 2 + 10^{(-i)}], [3], Z)$ given in Forrey's article.

Doing that in Maple however it is almost no problem (increase precision and switch to rationals in bad situations if needed). But it should be sayed, that it does not work fine for 'very large' parameters (the series would need long to converge, have not played with things like binary splitting or similar).

One could implement the stuff in a way, such that the command 'evalhf' could do the job in Maple (simplier for testing), but I wanted actual code. However the CodeGeneration package in Maple does not cover complex arithmetics.

The enclosed code compiles with MS VC2005 Express (had problems with Dev-C++), the interface for the DLL should also work with older Maple versions.

Some limitations and pitfalls, which I am aware of:

Essentially one can not expect really good results within given working precision as soon as one comes to close to situations, where one has to use the linear transformations from A&S and the parameters are in an exceptional constellation. The transformations always exclude certain combinations of the parameters to be an integer (for the usual transformations and for the exceptional cases as well).

But if they are only very close, then large rounding errors occure within the given working precision. It is easy to find that code pieces, they all check for 'isInteger', hence one could add some warning message. This also happens, if one has an 'exact' solution for the exceptional cases (so the lazyness of using numerical differentiation is a bearable sin).

One can improve a bit by using a better reciprocal Gamma function (taking more care 'around negative integers') or writing code for the series of the regularized hypergeometric function - but that gives not very much and does not heal the actual problem.

This could be overcome using a higher precision library (the code carries over, except the complex Gamma function used).

Generally the parameters should not be too large (say below absolute value <= 12 in size, faculty will leave needed precision even if a good solution for the Gamma function is used for double precision) compared to the working precision.

And, yes, error estimates are missing ...