Evaluating the hypergeometric function 2F1 using a quadratic transformation due to Potts and Snow

Reference:

Peter John Potts: Computable Real Arithmetic Using Linear Fractional Transformations,
Report, Department of Computing, Imperial College of Science, Technology and Medicine,
London, (June-1996). URL: http://citeseer.ist.psu.edu/potts96computable.html.

```
> restart; interface(version); Digits:=14:
  myFont:=[COURIER,10]:
  myPlotDefault:=
    thickness =0, font=myFont,axesfont=myFont,labelfont=myFont,titlefont=myFont, symbolsize=8:
```

Classic Worksheet Interface, Maple 12.02, Windows, Dec 10 2008 Build ID 377066

We take the following quadratic transformation (which is as in Abramowitz & Stegun, 15.3.23, p. 560):

```
> Snow:= z -> (sqrt(1-z) - 1)/(sqrt(1-z) + 1);
  ``;
  w =Snow(z);
  z = solve(%,z);
```

$$Snow := z \to \frac{\sqrt{1-z}-1}{\sqrt{1-z}+1}$$

$$w = \frac{\sqrt{1-z}-1}{\sqrt{1-z}+1}$$

$$z = -\frac{4\,w}{(w-1)^2}$$

On p.21 Potts refers to the book "Hypergeometric and Legendre Functions with Applications" (1952) by Chester Snow for the following 3 term recursion (however I was not able to locate it in the given reference, so I guess Potts invested some work here ... calling it Snow-Potts sound a bit silly), valid in the cut plane:

```
> hypergeom([a,b],[c],z) = (1-w)^a*Sum(h(n)*w^n, n=0..infinity);
  ``;
  h(0) = 1, h(1) = 2*a/c*(c-2*b);
  h(n+2) = (n+2*a)*(n+2*a+1-c)/(n+2)/(n+1+c)*h(n)+2*(c-2*b)*(n+1+a)*h(n+1)/(n+2)/(n+1+c);
```

$$\mathrm{hypergeom}([a, b], [c], z) = (1-w)^a \left(\sum_{n=0}^{\infty} h(n)\, w^n\right)$$

$$h(0) = 1,\ h(1) = \frac{2\,a\,(c-2\,b)}{c}$$

$$h(n+2) = \frac{(n+2\,a)\,(n+2\,a+1-c)\,h(n)}{(n+2)\,(n+1+c)} + \frac{2\,(c-2\,b)\,(n+1+a)\,h(n+1)}{(n+2)\,(n+1+c)}$$

That series converges for |w| < 1 (if z is purely real, then |Snow(z)| = 1).
I use that series, if |w| is not too large: the threshold will S0 = 8/9, see below, though I try to keep it below 1/2.

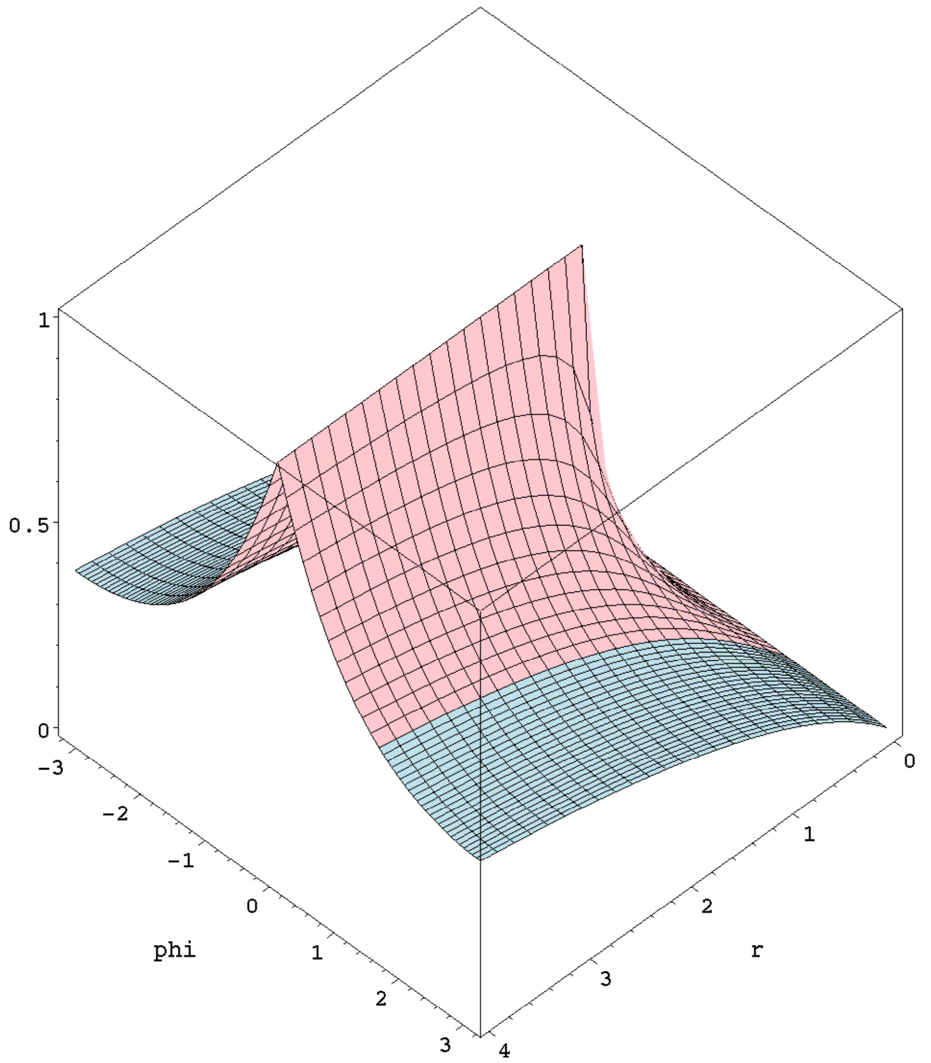
Let us look at the results of that quadratic transform

```
> abs(Snow(z)); r*exp(I*phi):
```

$$\left|\frac{\sqrt{1-z}-1}{\sqrt{1-z}+1}\right|$$

```
> plot3d(abs(Snow(r*exp(I*phi))), r=0 .. 4, phi=-Pi/2 .. Pi/2, myPlotDefault, axes=boxed,  color =
  "LightPink"):
  plot3d(abs(Snow(r*exp(I*phi))), r=0 .. 4, phi=-Pi .. -Pi/2, axes=boxed,                   color =
  "LightBlue"):
  plot3d(abs(Snow(r*exp(I*phi))), r=0 .. 4, phi=Pi/2 .. Pi, axes=boxed, tickmarks=[4,6,3], color =
  "LightBlue"):
  plots[display](%,%%,%%%,
    title="abs(Snow(z)) for z = r*exp(I*phi), red = z in right, blue = z in left half plane");
```

abs(Snow(z)) for z = r*exp(I*phi), red = z in right, blue = z in left half plane



So for Re(z) < 0 we always have |w| <= 1/2, w = Snow's variable, if we take |z| < 4 and for those towards '-infinity' one we can take the transform 1/z: that finally can be done through a Taylor series for 2F1 in 0 and that can be done well for a radius = R0, where R0 = 0.9 usually is fine.

One can even take the radius a bit larger in the left half plane using r = 40/9:

```
> 1/2= 'abs(Snow(r*exp(I*Pi/2)))'; %; evalc(%) assuming (0 < r): evala(%);
  r in {solve(%, r)}; #evalf(%);
```

$$\frac{1}{2} = \left|\mathrm{Snow}(r\,\mathbf{e}^{(1/2\,I\,\pi)})\right|$$

$$\frac{1}{2} = \left|\frac{\sqrt{1 - r\,I} - 1}{\sqrt{1 - r\,I} + 1}\right|$$

$$\frac{1}{2} = \sqrt{\frac{-2\sqrt{1+r^2}\,\sqrt{2\sqrt{1+r^2}+2} - 2\sqrt{2\sqrt{1+r^2}+2} + 4 + r^2 + 4\sqrt{1+r^2}}{r^2}}$$

$$r \in \{\frac{-40}{9}, \frac{40}{9}\}$$

Note that through that 2F1 already can be computed for the *complete* left half plane (up to exceptional parameter constellations).

```
> R0:=9/10;
```

$$R0 := \frac{9}{10}$$

For the right half plane the maximum (radius = abs(z) fixed is achieved in purely real values and desiring abs(z) = 1/2 gives a bound:

```
> 'abs(Snow(r*exp(I*0)))'=1/2; %;
  r in {solve(%,r)};
  ``;
  S0:=8/9; ``= evalf(%);
  'abs(Snow(S0*exp(I*0)))': '%'= evala(%);
```

$$\left|\mathrm{Snow}(r\,\mathbf{e}^{0})\right| = \frac{1}{2}$$

$$\left|\frac{\sqrt{1-r}-1}{\sqrt{1-r}+1}\right| = \frac{1}{2}$$

$$r \in \{-8, \frac{8}{9}\}$$

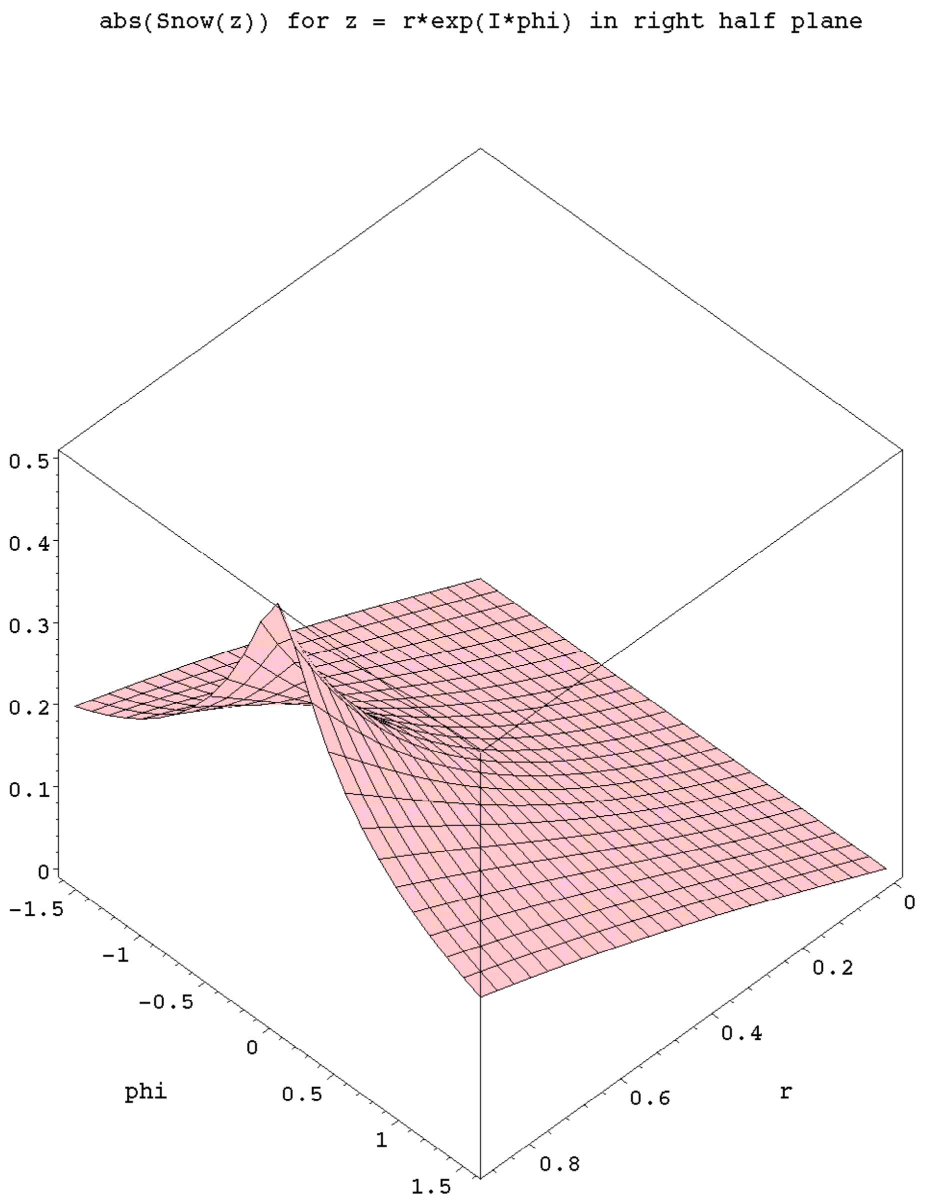
$$S0 := \frac{8}{9}$$

$$= 0.8888888888888889$$

$$\left|\mathrm{Snow}(S0\,\mathbf{e}^{0})\right| = \frac{1}{2}$$

```
> myRange:= 'r=0 .. S0, phi=-Pi/2 .. Pi/2';
  plot3d(abs(Snow(r*exp(I*phi))), myRange, myPlotDefault, axes=boxed, color = "LightPink",
    title="abs(Snow(z)) for z = r*exp(I*phi) in right half plane");
```

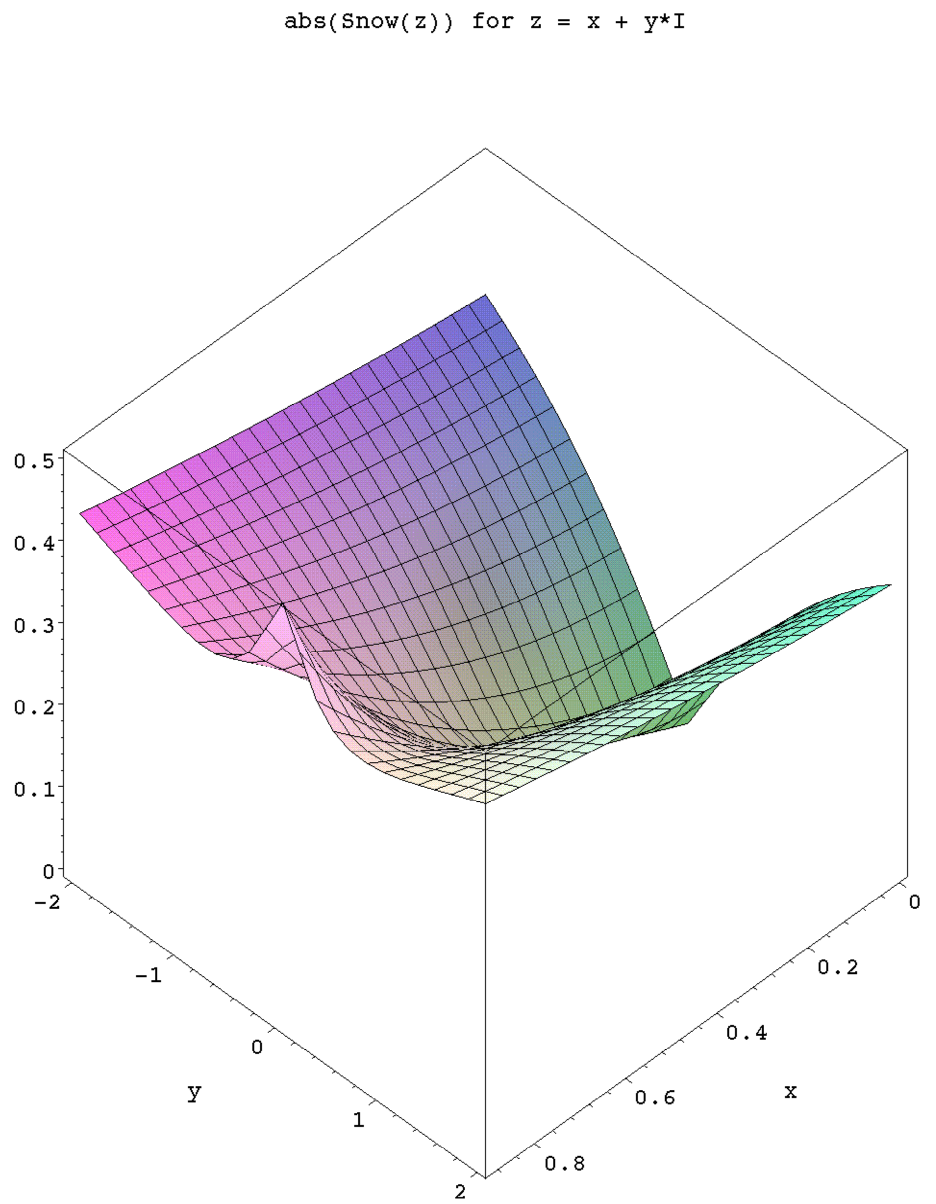
$$myRange := r = 0 \,..\, S0, \phi = -\frac{\pi}{2} \,..\, \frac{\pi}{2}$$



In cartesian coordinates one even has a nice rectangle, where Snow(z) <= 1/2 in size (and already covers the nasty z = exp(I*Pi/6) = diagonal intersecting the UnitCirclele)

```
> myRange:='x = 0 ..S0, y = -2 .. 2';
  plot3d(abs(Snow(x+I*y)), myRange, myPlotDefault, axes=boxed, title="abs(Snow(z)) for z = x + y*I");
```

myRange := x = 0 .. S0, y = -2 .. 2



Using 1/z if 4 < |z| the left half plane is completely done (in the linear transformations the Taylor series around 0 will be used).

For the right half plane one uses 1/z for 2 < |z|. Then two segments around the unit circle remain remain (see the graphics below), they are symmetric w.r.t. the x axis and the are treated in the rest of that note.

```
> UnitCircle:=exp(I*phi);
```

$$\mathrm{UnitCircle} := e^{(\phi I)}$$

```
> phi0:='phi0':
  #S0 = Re(cos(phi) + sin(phi)*I); evalc(%): solve(%, phi):
  'S0 = Re(exp(I*phi))'; evalc(%): solve(%, phi):
  phi0:=%;
  ``=evalf(%);
```

$$S0 = \Re(e^{(\phi I)})$$

$$\phi 0 := \arccos\left(\frac{8}{9}\right)$$

$$= 0.47588224966041$$

For ϕ larger than $\phi 0 = \arccos\left(\frac{8}{9}\right)$ a point on the circle will fall into the grey rectangle (see figure below), were Snow's method works. That covers the nasty point $e^{\left(\frac{I\pi}{3}\right)}$, which can not be solely reached through iterated linear transforms. Fine.

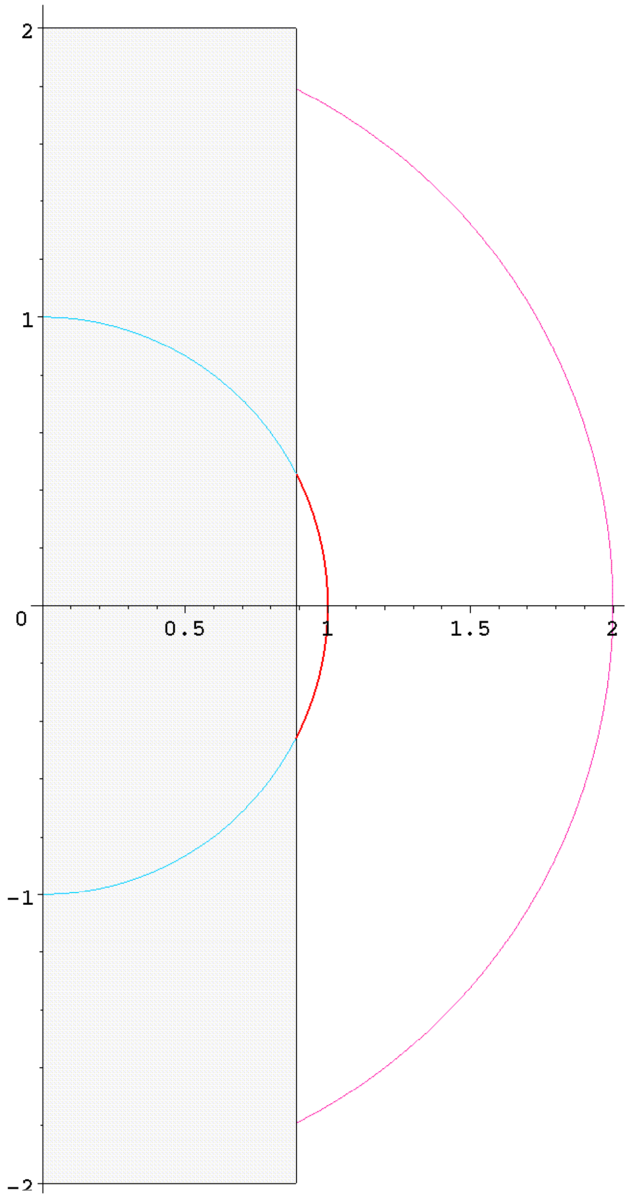
Using 1/z for 2 < |z| we also arrive in the grey rectangle (for points on that circle), if the angle is above $\phi 2$, given by the following condition:

```
> 'S0 = Re(2*exp(I*phi2))'; evalc(%): isolate(%,phi2); #evalf(%);
```

$$S0 = \Re(2\,\mathbf{e}^{(\phi 2\, I)})$$

$$\phi 2 = \arccos\left(\frac{4}{9}\right)$$

```
> plot([Re(UnitCircle),Im(UnitCircle), phi=-phi0 ... phi0], myPlotDefault, thickness=2, color=red):
  plot([Re(UnitCircle),Im(UnitCircle), phi=-Pi/2 ... +Pi/2], color="DeepSkyBlue"):
  plottools[rectangle]([0,-2],[S0, 2], color="WhiteSmoke"):
  plots[display](%%%,%%,%):

  plot([Re(2*UnitCircle),Im(2*UnitCircle), phi=-arccos(4/9) ... +arccos(4/9)], scaling=constrained,
  color="DeepPink"):
  plots[display](%%,%);
```



Between the grey rectangle and the exterior circle segment one can apply $z \rightarrow \frac{z-1}{z}$ (which is A&S 15.3.9, Paff's transformation followed by $\frac{1}{z}$).

But only for those points which end up in the 'numerical' radius $R0$ for the Taylor series around 0. For z towards 0 the transformed explodes, so one takes the closest point towards 0 in the region for which the transform still has to fine. That is z = $S0 + 0 * I$ and taking that as a minimal radius we get the needed angle:

```
> as9:= z -> (z-1)/z;
  ``;
  'R0 = eval(abs(as9(r*exp(I*phi))),r=S0)';
```

```
% assuming phi::real;
[solve(%, phi)]; evalf(%);
```

$$as9 := z \to \frac{z-1}{z}$$

$$R0 = \left| as9(r\,\mathbf{e}^{(\phi I)}) \right| \Big|_{r = S0}$$

$$\frac{9}{10} = \sqrt{\left(\cos(\phi) - \frac{9}{8}\right)^2 + \sin(\phi)^2}$$

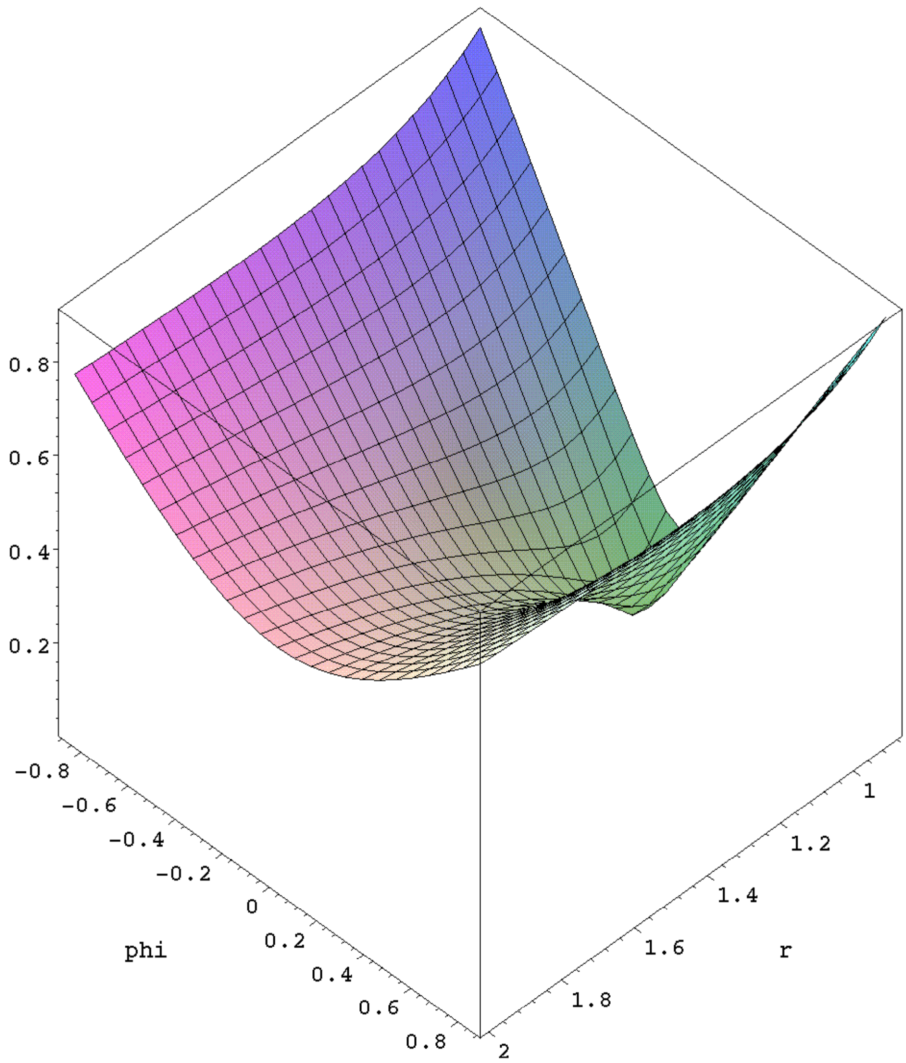
$$\left[\arctan\left(\frac{77\sqrt{1271}}{2329}\right), -\arctan\left(\frac{77\sqrt{1271}}{2329}\right)\right]$$

$$[0.86722582630957, -0.86722582630957]$$

Again just check through plotting the situation of applying $z \to \frac{z-1}{z}$ first and then using the Taylor series :

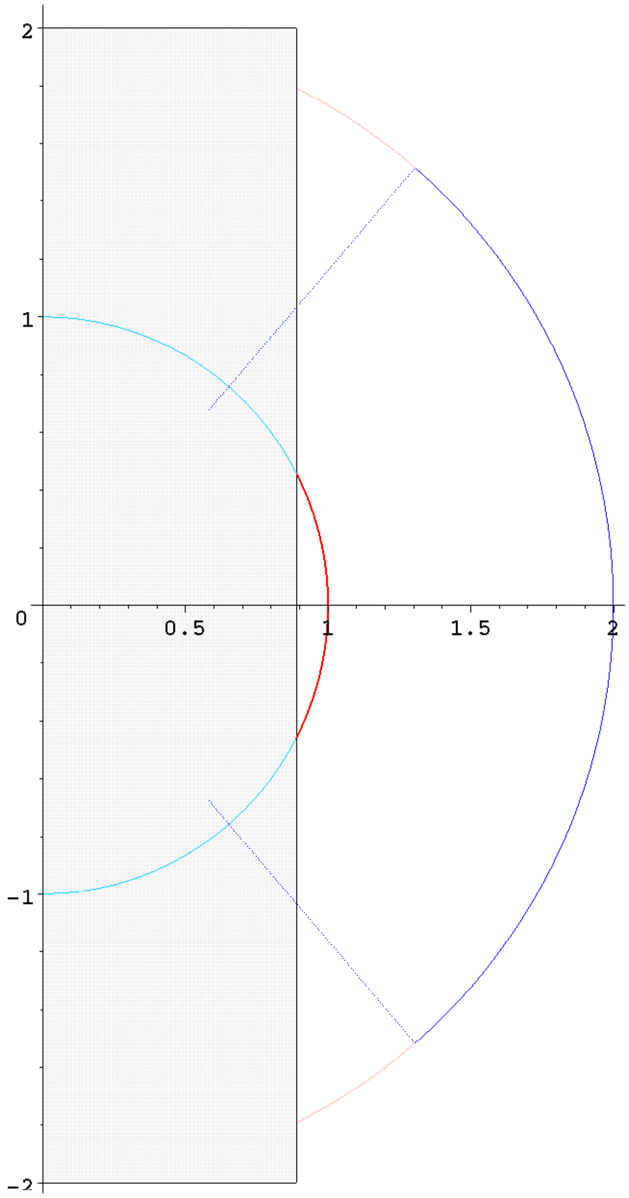
```
> 'abs(as9(r*exp(I*phi)))': '%'=% assuming phi::real; #min(2, %);
  plot3d(rhs(%), r = S0 .. 2, phi = -0.86 ... 0.86, axes=boxed, myPlotDefault);
```

$$\left| as9(r\,\mathbf{e}^{(\phi I)}) \right| = \left| \frac{-1 + r\,\mathbf{e}^{(\phi I)}}{r} \right|$$



Now we have covered almost all we need:

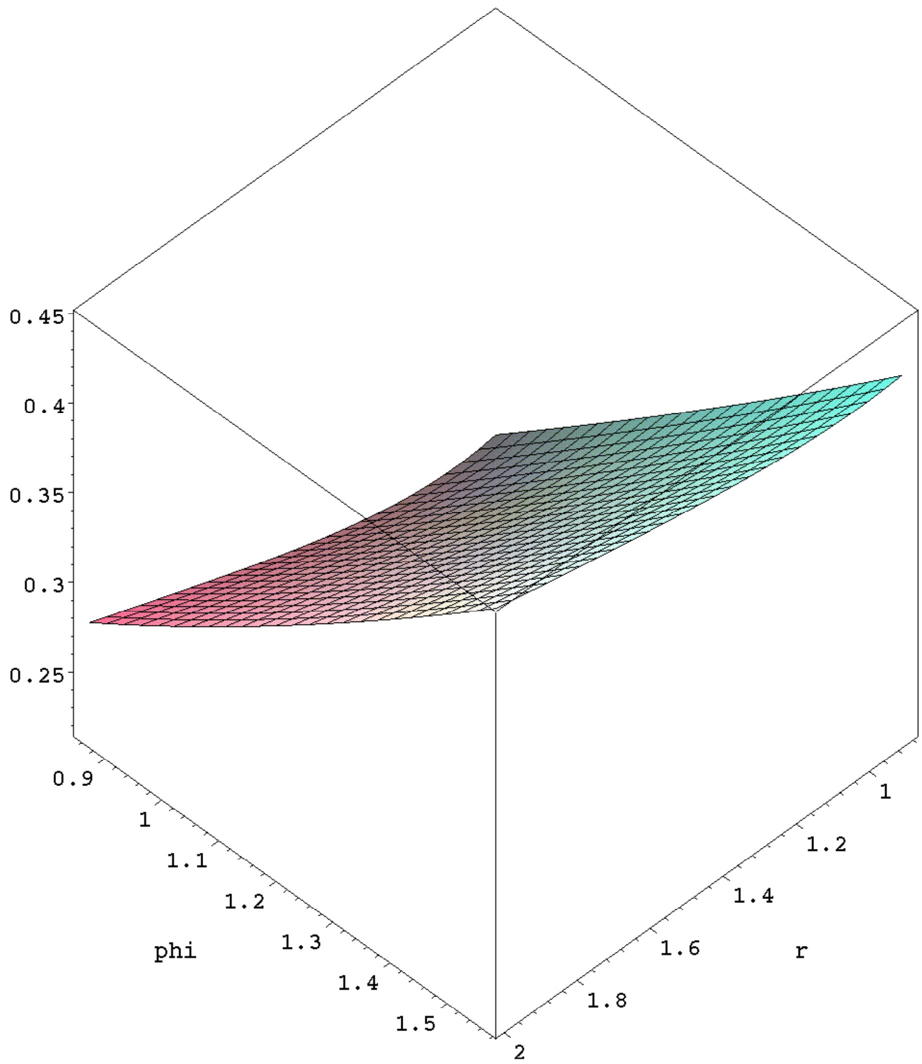
```
> plot([Re(UnitCircle),Im(UnitCircle), phi=-phi0 ... phi0], myPlotDefault, thickness=2, color=red):
  plot([Re(UnitCircle),Im(UnitCircle), phi=-Pi/2 ... +Pi/2], color="DeepSkyBlue"):
  plottools[rectangle]([0,-2],[S0, 2], color="WhiteSmoke"):
  P1:=plots[display](%%%,%%,%):

  plot([Re(2*UnitCircle),Im(2*UnitCircle), phi=-0.86 ... +0.86], scaling=constrained, color=blue):
  plot([Re(2*UnitCircle),Im(2*UnitCircle), phi=+0.86 ... +arccos(4/9)], thickness=2, color="Pink"):
  plot([Re(2*UnitCircle),Im(2*UnitCircle), phi=-arccos(4/9) ... -0.86], thickness=2, color="Pink"):
  P2:=plots[display](%%%,%%,%):

  plot([Re(r*exp(+I*0.86)),Im(r*exp(+I*0.86)), r=S0 ..2], linestyle=dot, color=blue):
  plot([Re(r*exp(-I*0.86)),Im(r*exp(-I*0.86)), r=S0 ..2], linestyle=dot, color=blue):
  P3:=plots[display](%%,%):

  plots[display](P1, P2, P3);
```



For the remaining region between the outer circle, the grey rectangle and the dotted radius one can use $z \to 1 - z$ to arrive at the case for Snow's series, the values will be small in magnitude:

```
> #r*UnitCircle;
  #1 - %;
  #1/%;
  'abs(Snow(1 - r*UnitCircle))': '%'= %; #evalc(%) assuming ( 0<r, phi::real);
  plot3d(rhs(%), r = S0 .. 2, phi = 0.86 ... Pi/2, axes=boxed, myPlotDefault);
```

$$\left|\mathrm{Snow}(1 - r\,\mathrm{UnitCircle})\right| = \left|\frac{\sqrt{r\,\mathbf{e}^{(\phi I)}} - 1}{\sqrt{r\,\mathbf{e}^{(\phi I)}} + 1}\right|$$



What do I (currently) miss?

a) The proof for the identity 2F1 = formally recursive series
b) Convergence proof and insight for numerical stability (thus keeping below 8/9 or even 1/2)

>
>