Numerical Methods Homework 2

J L Gouws

February 11, 2024

1. The source code is available here. I did it in a script as it was easier for me to work with. I used the following packages: Printf, LaTeXStrings, CairoMakie, HCubature, SphericalHarmonics, LegendrePolynomials, and DifferentialEquations.

   (a) I use the spherical harmonics package to determine the spherical harmonics and then hcubature to determine the coeficients of the function expansion.

```
expandSphericalHarmonics2(f, lₘₐₓ) = hcubature(
  x -> sin(x[1]) * f(x[1], x[2]) .* conj(flattenSHArray(computeYlm(x[1], x[2],
     lmax = lₘₐₓ)))
  , [0, 0], [π, 2 * π]
)
```

   Where I wrote a function to convert the output of spherical harmonic function into a flat array:

```
function flattenSHArray(shArray, lₘₐₓ)
  l = 0
  m = 0
  coeffVector::Array{ComplexF64} = []
  for a in 0:lₘₐₓ * (lₘₐₓ + 2)
    push!(coeffVector, shArray[(l, m)])
    if m == l
      l+=1
      m = -l
    else
      m+=1
    end
  end
  coeffVector
end
```

   (b) I will use these initial conditions for the wave equation:

```
  ϕ₀(θ, ϕ) = exp( - θ^2 / 0.4);
  ψ₀(θ, ϕ) = 0;
```

   And next I will detail the more technical details of the numerical solver. I wrote this function to differentiate an array of spherical harmonic coefficents from an expansion.

```
function LaplacianSphericalCoefficientsArrays(coeffs)
  l = 0
  m = 0
  coeffVector::Array{ComplexF64} = []
  for a in coeffs
    push!(coeffVector, - l * (l + 1) * a) #multiply coefficient by correct
     factor
    if m == l
      l+=1
```

```
      m = -l
    else
      m+=1
    end
  end
  coeffVector
end
```

Now we have all the ingredients to write a function that will solve the wave equation:

```
function waveSystem(U)
    half = Int(size(U)[1] // 2)
    ϕ = U[begin:half] #split the state vector in two
    ψ = U[half + 1:end]
    vcat(ψ, LaplacianSphericalCoefficientsArrays(ϕ)) #concatenate the two
end

function solveWaveEquation(ϕ₀, ψ₀, N, t₀, t₁)
  ϕ₀SphericalCoeffs = expandSphericalHarmonics2(ϕ₀, N)[1]; #get expanstion of
  ψ₀SphericalCoeffs = expandSphericalHarmonics2(ψ₀, N)[1]; # initial conditions

  alg = DP5()
  U0 = vcat( ϕ₀SphericalCoeffs, ψ₀SphericalCoeffs)         #I solve this with an
  prob = ODEProblem((U,p,t) -> waveSystem(U), U0, (t₀, t₁))# an integrator from
     a pacakage
  solve(prob, alg)
end
```

As can be seen from the above code I use a solver in the DifferentialEquations package. And do some accounting to keep track of the different arrays for ϕ and ψ. And lastly we need a function that will give us a function from these coefficients:

```
function functionFromSphericalCoefficients(coeffs)
  l = 0
  m = 0
  lₘₐₓ = 0
  for a in 1:size(coeffs)[1]
    lₘₐₓ = l
    if m == l
      l+=1
      m = -l
    else
      m+=1
    end
  end
  function f(θ, ϕ)
    harmonics = flattenSHArray(computeYlm(θ, ϕ, lmax = lₘₐₓ))
    harmonics' * coeffs
  end
end
```

(c) This part is now easy since we have all the ingredients.

```
  ϕ₀(θ, ϕ) = exp( - θ^2 / 0.4);
  ψ₀(θ, ϕ) = 0;
  sol = solveWaveEquation(ϕ₀, ψ₀, lmax, 0, 10)
```

And I use this solution to reconstruct the function for the solution.

```
sizA = Int(size(sol.u[1])[1] / 2)

firstprofileWhole = sol[1][begin:sizA]
myFuncFirst = functionFromSphericalCoefficients(firstprofileWhole)
```

(d) Below are figures of the solution at different times. The plots show the solutions of the wave equations as function of θ for the $\phi = \pi/2$. I found these graphs the easiest to interpret. I might have gone overboard with the testing–I did five different solutions for cut-offs $l_{\text{max}} = 2, 4, 6, 8, 10$. As Figure 1 shows, in particular the first subfigure, for $l_{\text{max}} = 2$, the expansion does not capture the profile of the initial conditions very well, but the problem can still be evolved and a solution obtained.. As Figure 2 shows, for $l_{\text{max}} = 4$, the motion of the wave is clearer, but there are some ripples in the solution that are not apparent in the true solution. The ripples present for the cut-off $l_{\text{max}} = 4$ slowly become smaller for larger cut-offs, which is indicative of a convergent solution. This covergence can be qualititatively seen by comparing Figures 3, 4 and 5, keeping in mind that the profiles are printed for slightly different times.

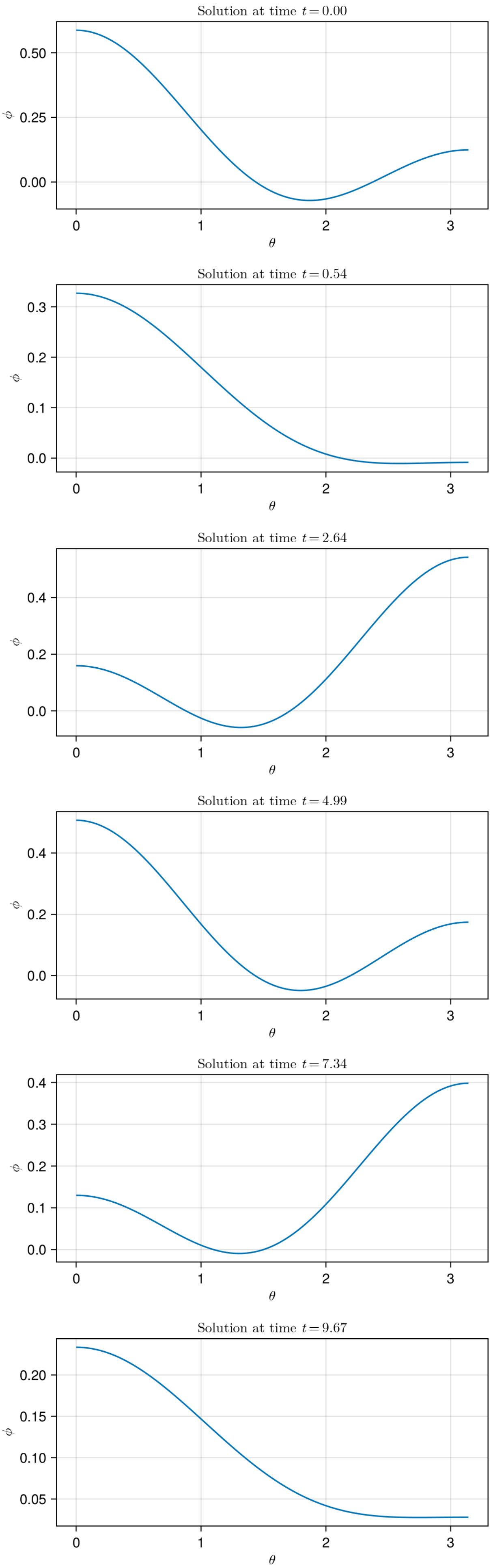

Figure 1: Solution of the wave equation with cut-off $l_{\max} = 2$

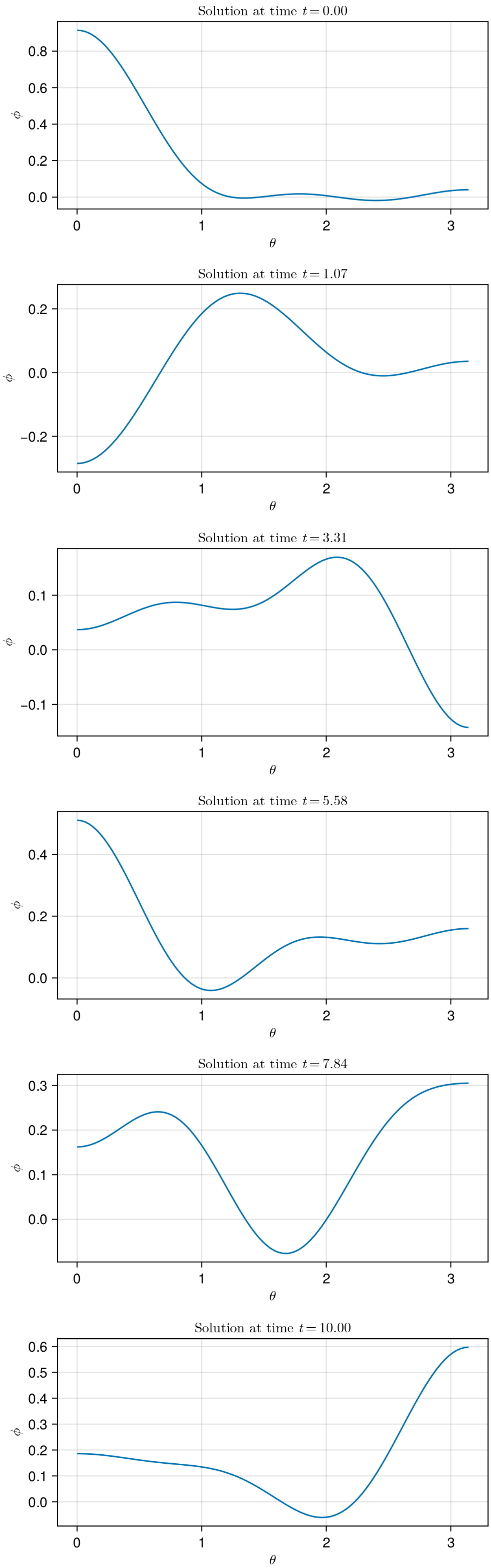

Figure 2: Solution of the wave equation with cut-off $l_{\max} = 4$

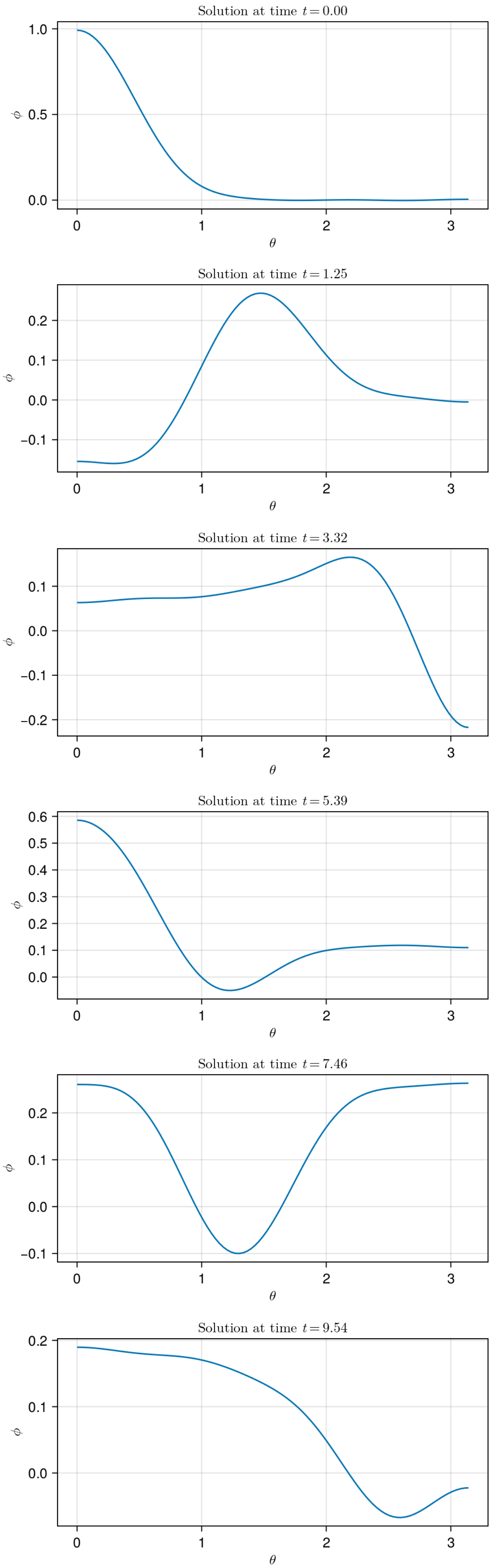

Figure 3: Solution of the wave equation with cut-off $l_{\max} = 6$

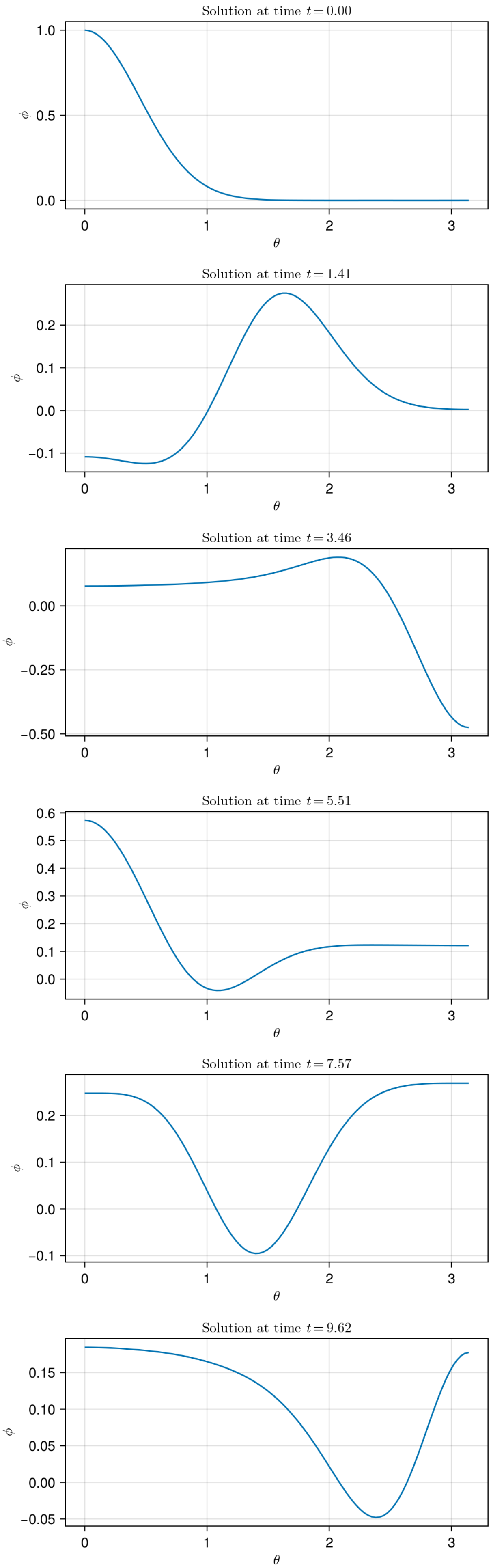

Figure 4: Solution of the wave equation with cut-off $l_{\max} = 8$

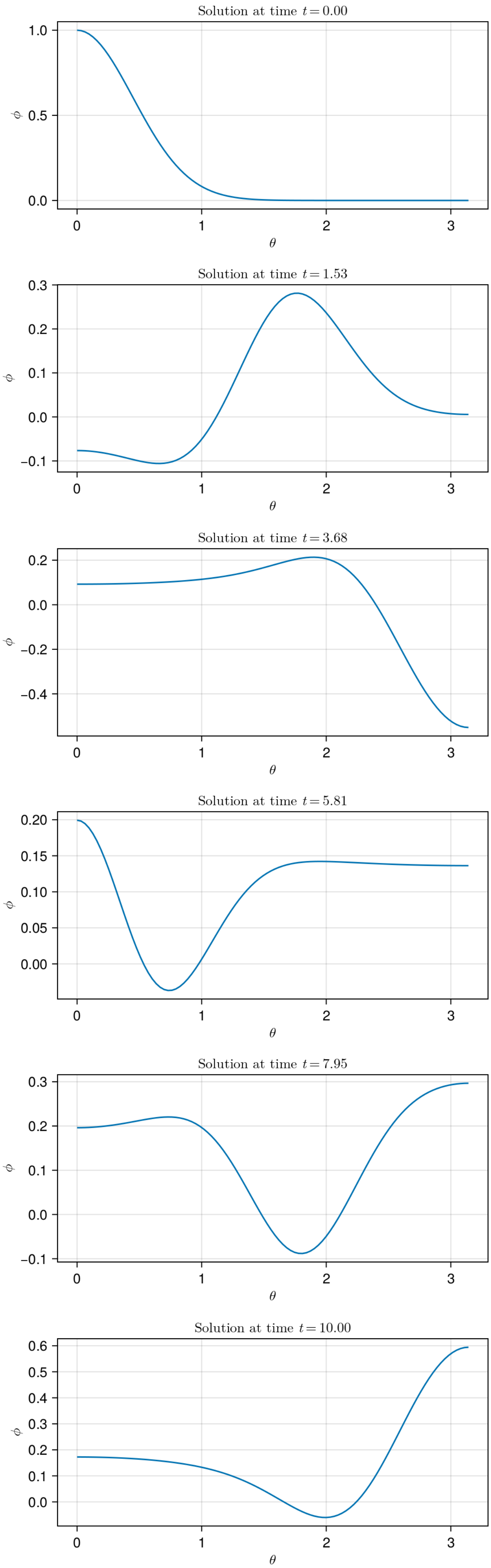

Figure 5: Solution of the wave equation with cut-off $l_{\max} = 10$

2. I also did the heat equation out of interest.

```
function heat(U)
  LaplacianSphericalCoefficientsArrays(U)
end

ϕ₀(θ, ϕ) = exp(- θ^2)
U0 = flattenSHArray(expandSphericalHarmonics2(ϕ₀, 15)[1])
prob = ODEProblem((U,p,t) -> heat(U), U0, (0.0, 10.0))

alg = Rodas5(autodiff = false)
solheat = solve(prob, alg);
```

Figure 6 shows the numerical solution of the heat equation with the heat dispersing over the sphere.

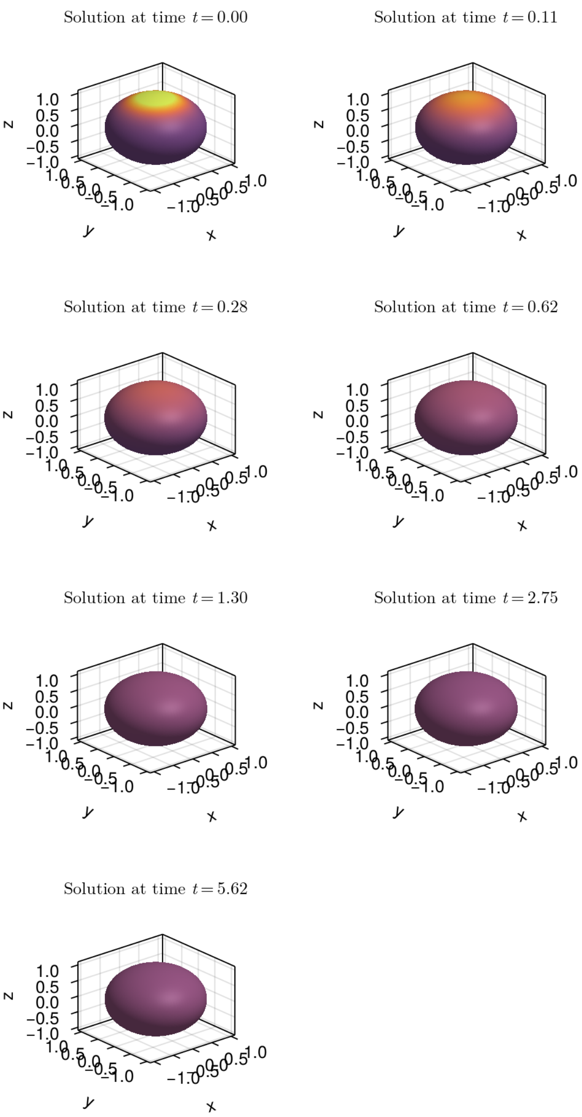

Figure 6: Solution of the heat equation with cut-off $l_{\max} = 15$