Evolution of Schwarzschild spacetime

Lorenzo Cipriani

September 27, 2022

(A.1) - Initial Data

In standard Schwarzschild coordinates, the line element of the spacial hypersufaces at $t = \text{cost}$ can be written as

$$\mathrm{d}\ell^2 = \frac{\mathrm{d}R^2}{1 - \frac{2M}{R}} + R^2\,\mathrm{d}\Omega^2$$

In isotropic coordinates the spacial part of the metric is conformally flat and it is written instead as

$$\mathrm{d}\ell^2 = \psi^4\left(\mathrm{d}r^2 + r^2\,\mathrm{d}\Omega^2\right), \quad \psi = 1 + \frac{M}{2r} \tag{1}$$

The relation between the two radial variables R and r can be found noticing that the two line elements describe the same spacetime. It is therefore true that

$$\begin{cases} \psi^4 r^2\,\mathrm{d}\Omega^2 = R^2\,\mathrm{d}\Omega^2 \\ \psi^4\,\mathrm{d}r^2 = \frac{\mathrm{d}R^2}{1-\frac{2M}{R}} \end{cases}$$

Dividing the second equation by the first and taking the squared root, the following differential equation is obtained:

$$\frac{\mathrm{d}r}{r} = \frac{\mathrm{d}R}{\sqrt{R^2 - 2MR}}$$

This equation is solved by separation of variables imposing the initial condition $R(\frac{M}{2}) = 2M$. The relation obtained is

$$R(r) = 2M\cosh^2\left[\frac{1}{2}\log\left(\frac{M}{2r}\right)\right] \tag{2}$$

From the definition of cosh in terms of exponential functions it is possible to easily cast Eq. (2) into the much more manageable form

$$R(r) = 2M\frac{(M+2r)^2}{8Mr} \equiv r\left(1 + \frac{M}{2r}\right)^2$$

that is the relation to be recovered.

Code implementation

The coding language chosen to complete the exercise is Python.
To solve the system of equations that represent Einstein's Field Equations in ADM formulation it has been employed the *Method of Lines*. The PDEs reported in Eqs. (3) of the problem set have been reduced to a set of ODEs discretizing the radial derivatives using 4th order finite-differencing stencils. The time integration has been performed using a 4th order Runge-Kutta method with a fixed time step.
The divergence of the initial data in $r = 0$ has been cured using both a staggered grid and the static puncture regularization of the fields A, B, D_A and D_B.
Taking advantage of the symmetries, each field has been simulated in only the radial direction on a segment of total length R. This domain has been discretized using N points; two ghost cells have been added to the inner boundary near $r = 0$ to impose the suggested symmetry conditions. The outer boundary condition is instead applied considering two other ghost cells whose value is updated using a simple first order extrapolation from the inside of the simulation. Since however the simulation domain is chosen to be much larger than the region in which the dynamics happens, this choice is practically equivalent to imposing Dirichlet boundary condition, keeping the fields fixed to their initial value.
The increased cost of performing the necessary steps for the RK4 algorithm instead of a simpler Euler method has been dampened using *NumPy* vectorized operations to compute both the right hand side of each equation and the derivatives. A further speed-up in the execution has been obtained using the *Numba* package, which translates Python functions to optimized machine code at runtime. The combination of these two packages allows to set the parameter N up to $\sim 10^4$ and to perform in less than ten minutes up to $\sim 10^5$ time steps.
In particular the time needed to perform one complete Rung-Kutta step decreased from $t_{step} > 1$s to $t_{step} = 5.49$ms $\pm$ 337µs and finally to $t_{step} = 3.94$ms $\pm$ 243µs. These times have been estimated using the magic command %%timeit in a Jupyter Notebook.
The horizon position has been found by looking for the zeros of Eq. (13) in the problem set: the definition of the expansion has been evaluated on all the points of the grid; a simple algorithm has been used to find the outermost point in which the array entries change sign. This crude approximation has been improved using the package *SciPy*: the four nearest grid points have been interpolated using a cubic spline and Eq. (13) is solved again using the function *fsolve*.

(A.2) - ADM evolution with geodesic slicing

Throughout all the exercise the mass parameter has been chosen to be $M = 1$. The simulation shown here has been performed using a uniform grid of length $R = 10M$ with 3000 points; in total 31001 time steps have been performed to reach a final time of $t = 3.1M$.

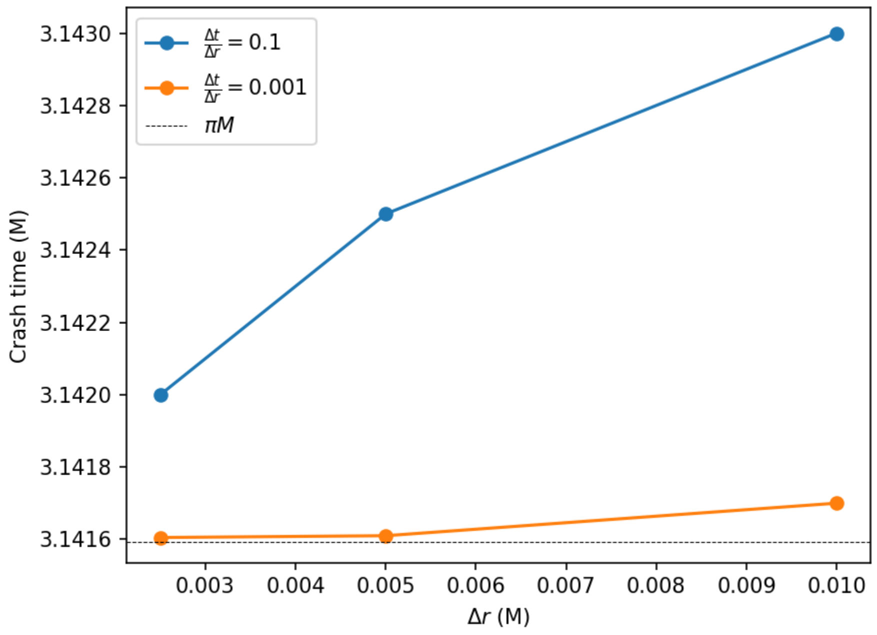
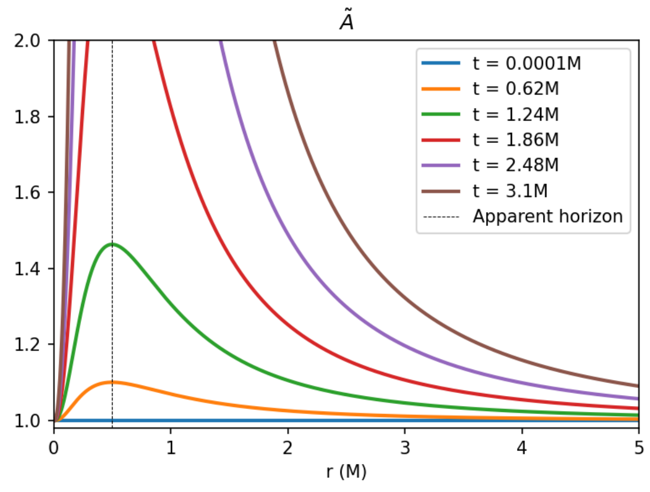
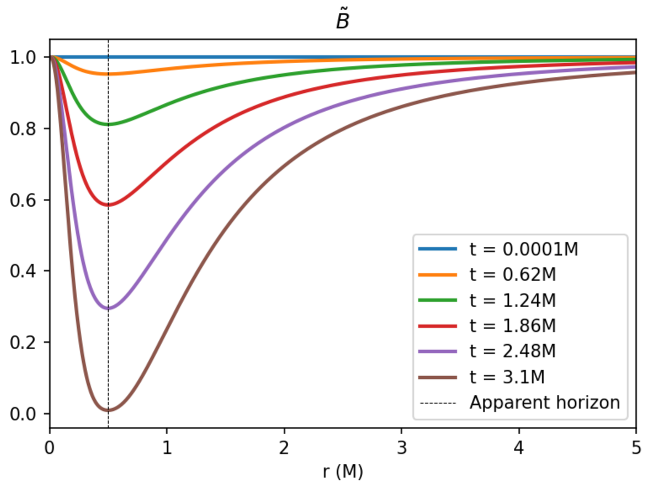


Figure 1: Times of code crash. The analytic result is plotted as a dashed line. The hardware used to perform the simulations is a laptop equipped with an 11th Generation Intel Core i7-11800H and 16GB of RAM.

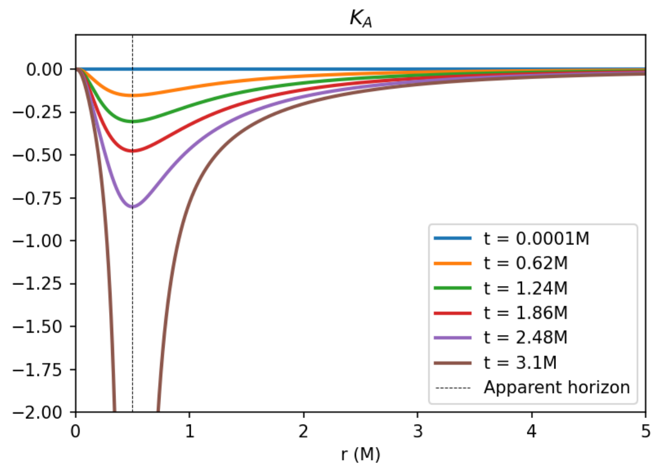
The four relevant fields - $\tilde{A}$, $\tilde{B}$, K_A and K_B - have been plotted in Fig. 2 employing at first the Geodesic Slicing gauge condition; the last two figures represent the apparent horizon variation in time and the apparent surface of the black hole.
Though the evolution starts from a static and isotropic configuration the system is quickly led away from isotropy into a dynamical evolution.
This can be easily seen from the radial and angular part of the metric, Fig. 2a and 2b, where the former starts growing a peak centred on the horizon at $r = \frac{M}{2}$ and the latter start decreasing toward zero at the same value of the radial coordinate. This behaviour is expected: since the shift vector is fixed to zero, the coordinate system is linked to observers that fall in the black hole from different distances and thus with different accelerations; the distance between them is therefore bound to increase.
At large values of r both $\tilde{A}$ and $\tilde{B}$ tend to 1, so that the actual line element tends to that of flat spacetime, as expected since also $\psi \to 1$.
It is important to notice that if the Schwarzschild black hole is evolved in time using geodesic slicing, the code will crash after a finite - proper and coordinate - time. In Fig. 1 there are plotted the crash times for two sets of simulations in which the Courant factor has been kept constant at $\Delta t/\Delta r = 0.1$ and $\Delta t/\Delta r = 0.001$. What is found is that when the spacial resolution is increased and the time step decreased accordingly, the crash time approaches the value of π. This is not surprising since the time of free fall for an observed that starts at a distance from the origin equal to the Schwarzschild radius is exactly equal to πM. The improvement in the resolution of this



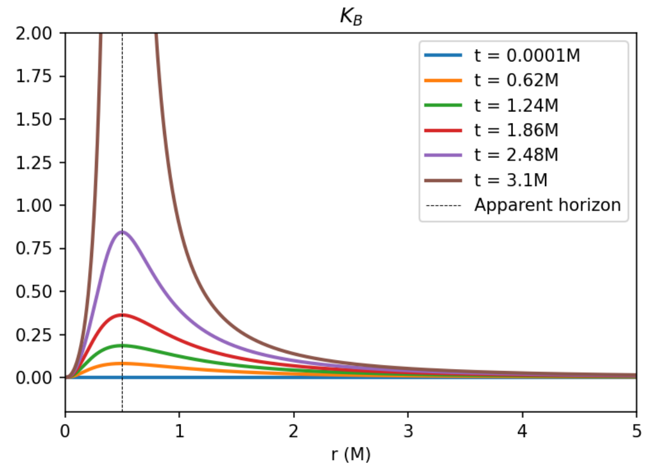
(a) $\tilde{A} = A/\psi^4$



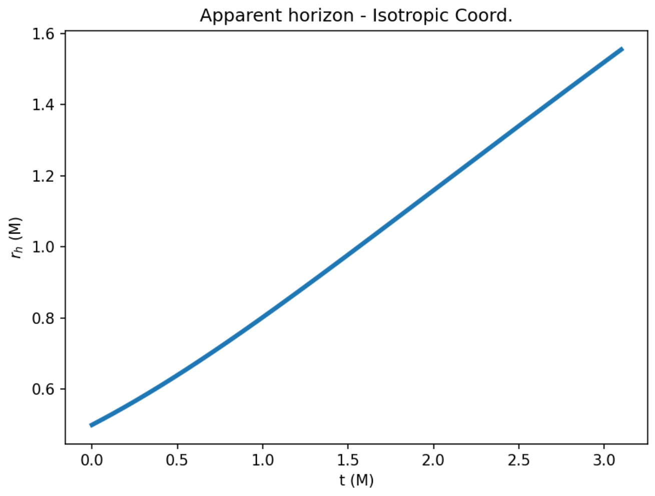
(b) $\tilde{B} = B/\psi^4$



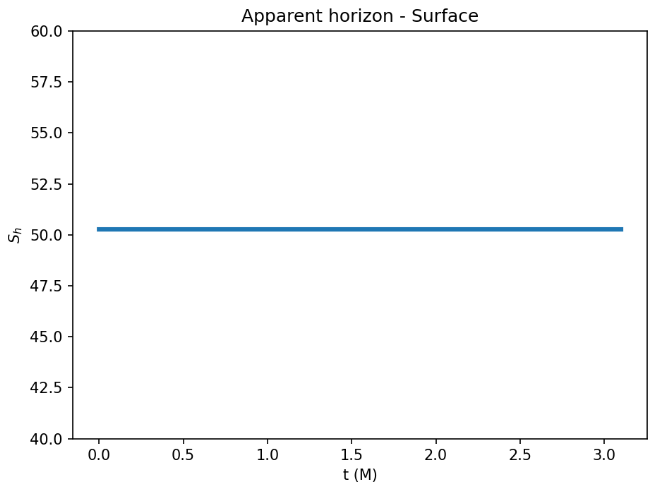
(c) $K_A = K^r{}_r$



(d) $K_B = K^\theta{}_\theta = K^\phi{}_\phi$



(e) Evolution in time of the apparent horizon



(f) Evolution in time of the apparent surface

Figure 2: Results of the simulation at different times for $\alpha = 1 =$ cost. The black dotted line represents the position of the event horizon at $r = \frac{1}{2}$. ψ is the same field defined in (1).

time is due to two effects: as the spacial resolution increases the throat of the black hole at $r = \frac{M}{2}$ is more clearly resolved; similarly, as the time step is reduced it is possible to better resolve the time at which the slice at constant time hits the singularity.
It is clear from Fig. 2a, 2b, 2c and 2d that the Schwarzschild metric is not static in this gauge. Another proof of this comes from Fig. 2e, where the radius of the apparent horizon of the black hole is plotted against time. The radius r_h has been computed looking for the outermost zero of Eq. (13) in the problem set.
At the beginning ($t = 0$) r_h is exactly equal to $M/2$, as expected from the isotropic initial conditions; during the evolution it starts to grow, reaching the value of $r_h \sim 1.5$ at $t = 3M$.
This is not a physical effect, but one linked to the fact that the radial coordinate follows free falling observers. This is also supported by Fig. 2f, that shows the surface of the apparent horizon: it remains constant throughout the whole the simulation to a value that closely resembles the theoretical one of $S_h^{Th} = 16\pi \approx 50.2$.

(A.3) $1 + \log$ Slicing

The gauge condition is now changed from $\alpha = 1$ to the so called *1+log* slicing, in which the evolution equations for the fields α and $D_\alpha = \partial_r \log \alpha$ becomes

$$\begin{aligned} \partial_t \alpha &= -\alpha^2 f(\alpha)(K_A + 2K_B) \\ \partial_t D_\alpha &= -\partial_r \left[\alpha f(\alpha)(K_A + 2K_B)\right] \end{aligned} \tag{3}$$

where in particular the arbitrary function f is chosen to be $f(\alpha) = 2/\alpha$.
The simulation is now performed using a uniform grid of length $R = 10M$ with 4000 points and 62801 time steps, in order to achieve a total simulation time of $t = 6.28M$. This value of the maximum time has been chosen to show all the relevant phases of the evolution without reaching the crash of the code.
The evolution of the dynamical fields $\tilde{A}$, $\tilde{B}$, K_A and K_B is shown in Fig. 4; it follows qualitatively that of geodesic slicing up until a certain point in time where the lapse function - that now is not constant any more, see Fig. 4e - reaches zero at the throat of the black hole. This causes the fields to stop evolving at that point; everywhere else the lapse keeps collapsing toward zero trying to avoid the singularity.
From that moment on time stops running on the horizon and at later times all around it, but α will be kept equal to one both at $r \to \infty$ and at the puncture $r = 0$ - that can be interpreted as the point $r' \sim \frac{1}{r} \to \infty$, the spacial infinity of another universe connected via the black hole's throat to the first -, causing large gradients that will lead to the crash of the code.
Unlike before, the crash time does not approach some specific value; increasing the spacial resolution of the code will increase the maximum value, even reaching $t \sim 10^2 \div 10^3 M$ when the right

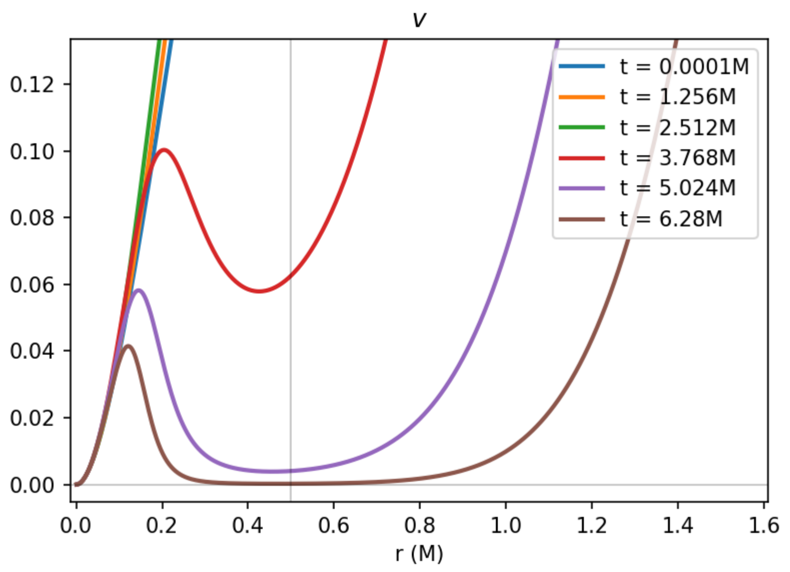


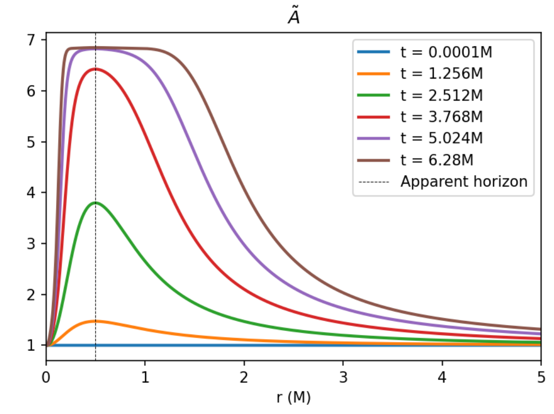
Figure 3: Velocity of propagation of information in the α field. For large values of r the velocity tends to $\sqrt{2}$.

combinations of time step and space discretization are chosen. Of particular interest is the fact that at low resolution - $\Delta r \gtrsim 0.015$ - the crash times coincide for the different Courant factors. The analysis of the propagation's speed of the information given by the evolution equation of α is also very informative, as it sheds light on why the gradients grow so strongly.
Since the gauge condition Eq. (3) is an hyperbolic equation, the speed is given by[1]

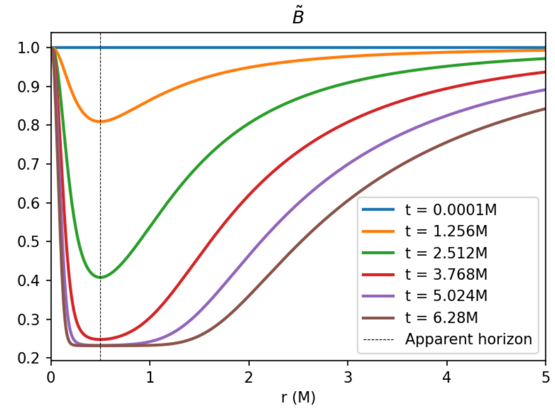
$$v = \alpha \frac{\sqrt{2\tilde{A}}}{\psi^2}$$

so when $r \to 0$, ψ diverges and also $v \to 0$, reflecting the fact that the puncture is infinitely far away; this behaviour is also recovered from the simulation, as shown in Fig. 3, where only the relevant part of the spacial domain is plotted. Note that even if in the asymptotic region $v \to \sqrt{2} > 1$ in units of the speed of light, this is not a problem since this is not a physical speed but the speed of propagation of the coordinate system.
The apparent horizon location is now less well behaved, but shows the same trend as before - see Fig 4f -. The horizon surface, on the other hand, remains constant to $S_h \approx 50.2$ once again, mirroring Fig. 2f.

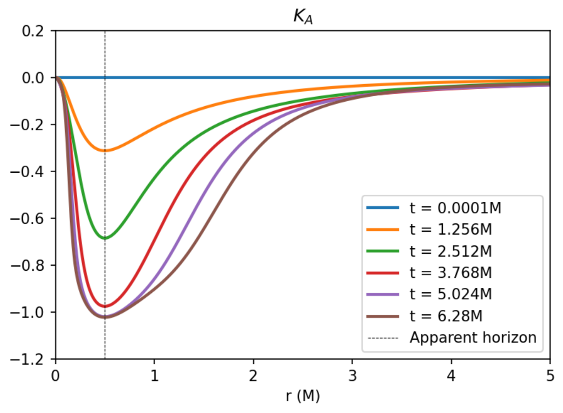
[1]The expression for the velocity is obtained differentiating the first of Eqs. (3) with respect to time and substituting the terms $\partial_t K_i$ with their evolution equation. α satisfies a wave equation with a well defined speed.



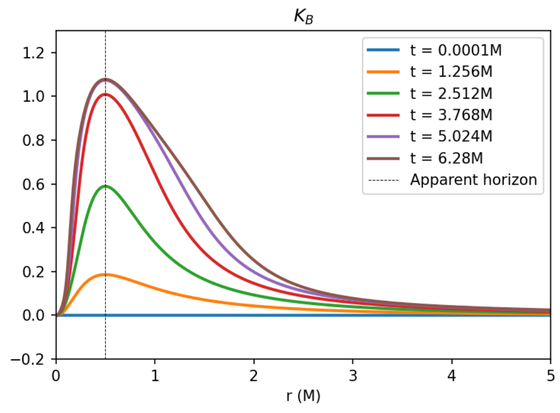
(a) $\tilde{A} = A/\psi^4$



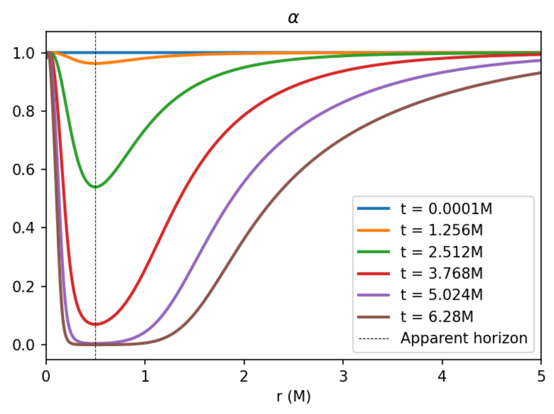
(b) $\tilde{B} = B/\psi^4$



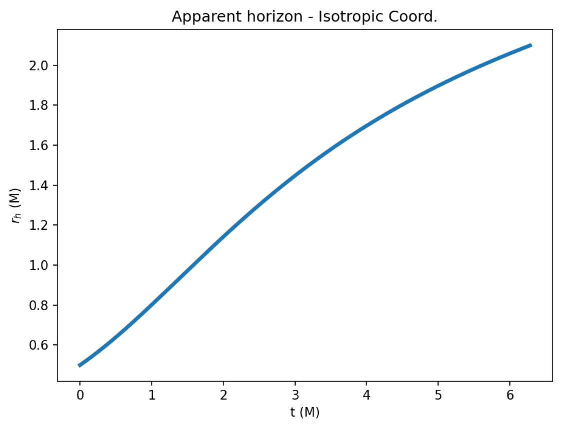
(c) $K_A = K^r{}_r$



(d) $K_B = K^\theta{}_\theta = K^\phi{}_\phi$



(e) Lapse function



(f) Apparent horizon variation in time

Figure 4: Results of the simulation at different times for $\partial_t \alpha = 2\alpha K$. The black dotted line represents the position of the event horizon at $r = \frac{1}{2}$. ψ is the same field defined in (1). The horizon surface associated with each r_h has not been reported since it mirrors the plot of Fig. 2f.