```
(*calculate the equation of photon motion first*)
(*upper index metric, lower index momentum*)
ClearAll["Global`*"];
清除全部
gij = {{-((a^2 + r^2)^2 - a^2 (a^2 - 2 r + r^2) Sin[θ]^2)/((r^2 + a^2 Cos[θ]^2) (a^2 - 2 r + r^2)), 0, 0, -(2 a r)/((r^2 + a^2 Cos[θ]^2) (a^2 - 2 r + r^2))},
   {0, (a^2 - 2 r + r^2)/(r^2 + a^2 Cos[θ]^2), 0, 0}, {0, 0, 1/(r^2 + a^2 Cos[θ]^2), 0},
   {-(2 a r)/((r^2 + a^2 Cos[θ]^2) (a^2 - 2 r + r^2)), 0, 0, ((a^2 - 2 r + r^2) - a^2 Sin[θ]^2)/((r^2 + a^2 Cos[θ]^2) (a^2 - 2 r + r^2) Sin[θ]^2)}};
p = {pt, pr, pθ, pϕ};
H = 1/2 (Sum[gij[[i, j]] × p[[i]] × p[[j]], {i, 1, 4}, {j, 1, 4}]);
        求和

du = {td, rd, θd, ϕd, prd, pθd};
td = D[H, pt] // Simplify;
     偏导        化简
rd = D[H, pr] // Simplify;
     偏导        化简
θd = D[H, pθ] // Simplify;
     偏导        化简
ϕd = D[H, pϕ] // Simplify;
     偏导        化简
prd = -D[H, r] // Simplify;
       偏导       化简
pθd = -D[H, θ] // Simplify;
       偏导       化简
Deom = du /.
  {pt → -EE, pϕ → LL, t → t[λ], r → r[λ], θ → θ[λ], ϕ → ϕ[λ], pr → pr[λ], pθ → pθ[λ]}
Print["--------------------------------Done
打印
   1------------------------------------------"]
```

Out[•]=

$$\Big\{-\frac{-a^4\,EE+2\,a\,LL\,r[\lambda]-2\,a^2\,EE\,r[\lambda]^2-EE\,r[\lambda]^4+a^2\,EE\left(a^2+(-2+r[\lambda])\,r[\lambda]\right)\mathrm{Sin}[\theta[\lambda]]^2}{\left(a^2+(-2+r[\lambda])\,r[\lambda]\right)\left(a^2\,\mathrm{Cos}[\theta[\lambda]]^2+r[\lambda]^2\right)},$$

$$\frac{pr[\lambda]\left(a^2+(-2+r[\lambda])\,r[\lambda]\right)}{a^2\,\mathrm{Cos}[\theta[\lambda]]^2+r[\lambda]^2},\ \frac{p\theta[\lambda]}{a^2\,\mathrm{Cos}[\theta[\lambda]]^2+r[\lambda]^2},$$

$$\frac{-a\,(a\,LL-2\,EE\,r[\lambda])+LL\,\mathrm{Csc}[\theta[\lambda]]^2\left(a^2+(-2+r[\lambda])\,r[\lambda]\right)}{\left(a^2+(-2+r[\lambda])\,r[\lambda]\right)\left(a^2\,\mathrm{Cos}[\theta[\lambda]]^2+r[\lambda]^2\right)},$$

$$\Big(-a^2\,\mathrm{Cos}[\theta[\lambda]]^2\Big(2\,a^3\,EE\,LL-2\,a\,EE\,LL\,r[\lambda]^2+pr[\lambda]^2\,(-2+r[\lambda])^2\,(-1+r[\lambda])\,r[\lambda]^2-$$
$$EE^2\,(-3+r[\lambda])\,r[\lambda]^4+a^4\left(pr[\lambda]^2\,(-1+r[\lambda])-EE^2\,(1+r[\lambda])\right)+$$
$$a^2\,(-1+r[\lambda])\left(LL^2+2\,r[\lambda]\left(pr[\lambda]^2\,(-2+r[\lambda])-EE^2\,r[\lambda]\right)\right)\Big)+$$
$$r[\lambda]\Big(-a^6\,EE^2-a^4\,LL^2+a^6\,pr[\lambda]^2+a^4\,p\theta[\lambda]^2+3\,a^4\,EE^2\,r[\lambda]+2\,a^3\,EE\,LL\,r[\lambda]+$$
$$3\,a^2\,LL^2\,r[\lambda]-5\,a^4\,pr[\lambda]^2\,r[\lambda]-4\,a^2\,p\theta[\lambda]^2\,r[\lambda]-2\,a^4\,EE^2\,r[\lambda]^2-8\,a\,EE\,LL\,r[\lambda]^2-$$
$$2\,a^2\,LL^2\,r[\lambda]^2+8\,a^2\,pr[\lambda]^2\,r[\lambda]^2+2\,a^4\,pr[\lambda]^2\,r[\lambda]^2+4\,p\theta[\lambda]^2\,r[\lambda]^2+$$
$$2\,a^2\,p\theta[\lambda]^2\,r[\lambda]^2+2\,a^2\,EE^2\,r[\lambda]^3+6\,a\,EE\,LL\,r[\lambda]^3-4\,pr[\lambda]^2\,r[\lambda]^3-6\,a^2\,pr[\lambda]^2\,r[\lambda]^3-$$
$$4\,p\theta[\lambda]^2\,r[\lambda]^3-a^2\,EE^2\,r[\lambda]^4+4\,pr[\lambda]^2\,r[\lambda]^4+a^2\,pr[\lambda]^2\,r[\lambda]^4+p\theta[\lambda]^2\,r[\lambda]^4-$$
$$EE^2\,r[\lambda]^5-pr[\lambda]^2\,r[\lambda]^5+LL^2\,\mathrm{Csc}[\theta[\lambda]]^2\left(a^2+(-2+r[\lambda])\,r[\lambda]\right)^2+$$
$$a^2\,EE^2\left(a^2+(-2+r[\lambda])\,r[\lambda]\right)^2\mathrm{Sin}[\theta[\lambda]^2]\Big)\Big)\Big/$$
$$\left(\left(a^2+(-2+r[\lambda])\,r[\lambda]\right)^2\left(a^2\,\mathrm{Cos}[\theta[\lambda]]^2+r[\lambda]^2\right)^2\right),$$

$$\Big(a^2\,LL^2\,\mathrm{Cot}[\theta[\lambda]]^3\left(a^2+(-2+r[\lambda])\,r[\lambda]\right)+\frac{1}{2}\,LL^2\,\mathrm{Cot}[\theta[\lambda]]\,\mathrm{Csc}[\theta[\lambda]]^2$$
$$\left(a^2+(-2+r[\lambda])\,r[\lambda]\right)\left(-a^2+a^2\,\mathrm{Cos}[2\,\theta[\lambda]]+2\,r[\lambda]^2\right)-a^2\,\mathrm{Cos}[\theta[\lambda]]$$
$$\Big(a^4\,pr[\lambda]^2+4\,a\,EE\,LL\,r[\lambda]+a^2\left(-LL^2+p\theta[\lambda]^2-2\,EE^2\,r[\lambda]+2\,pr[\lambda]^2\,(-2+r[\lambda])\,r[\lambda]\right)+$$
$$r[\lambda]\left(p\theta[\lambda]^2\,(-2+r[\lambda])+r[\lambda]\left(pr[\lambda]^2\,(-2+r[\lambda])^2-2\,EE^2\,r[\lambda]\right)\right)\Big)\,\mathrm{Sin}[\theta[\lambda]]\Big)\Big/$$
$$\left(\left(a^2+(-2+r[\lambda])\,r[\lambda]\right)\left(a^2\,\mathrm{Cos}[\theta[\lambda]]^2+r[\lambda]^2\right)^2\right)\Big\}$$

--------------------------------Done 1---

```
(*Kerr metric*)
ClearAll[MetricDown];
清除全部
MetricDown[a_][{t_, r_, θ_, ϕ_}] := Module[{tt, rr, θθ, ϕϕ, tϕ, Σ, Δ},
                                      模块
   Σ = r^2 + a^2 Cos[θ]^2;
   Δ = r^2 - 2 r + a^2;
   tt = - (1 - (2 r) / Σ);
   rr = Σ / Δ;
   θθ = Σ;
   ϕϕ = (r^2 + a^2 + (1 / Σ) × 2 a^2 r Sin[θ]^2) Sin[θ]^2;
   tϕ = - (1 / Σ) × 2 a r Sin[θ]^2;
   ( tt  0   0   tϕ
     0   rr  0   0
     0   0   θθ  0
     tϕ  0   0   ϕϕ )];

ClearAll[MetricUp];
清除全部
```

```
清除全部
MetricUp[a_][{t_, r_, θ_, ϕ_}] := Block[{tt, rr, θθ, ϕϕ, tϕ, Σ, Δ},
                                  块
    Σ = r^2 + a^2 Cos[θ]^2;
    Δ = r^2 - 2 r + a^2;
    tt = -1 - (2 r (r^2 + a^2))/(Δ Σ);
    rr = Δ / Σ;
    θθ = 1 / Σ;
    ϕϕ = (Δ - a^2 Sin[θ]^2)/(Δ Σ Sin[θ]^2);
    tϕ = -2 a r/(Δ Σ);
    {{tt, 0, 0, tϕ},
     {0, rr, 0, 0},
     {0, 0, θθ, 0},
     {tϕ, 0, 0, ϕϕ}}];
(*change θ to 0~π, change ψ to 0~2π*)
ClearAll[WrapAngle];
清除全部
WrapAngle[a1_, a2_] := Module[{x1 = a1, x2 = a2},
                       模块
    While[x1 > π, x1 = x1 - 2 π];
    While循环
    While[x1 < -π, x1 = x1 + 2 π];
    While循环
    If[x1 < 0, x1 = -x1; x2 = x2 + π];
    如果
    While[x2 ≥ 2 π, x2 = x2 - 2 π];
    While循环
    While[x2 < 0, x2 = x2 + 2 π];
    While循环
    {x1, x2}];
(*ZAMO*)
ClearAll[LocalTetrads];
清除全部
LocalTetrads[pos_, metric_] := Module[{e0, e1, e2, e3, gtt, grr, gθθ, gϕϕ, gtϕ},
                               模块
    {gtt, grr, gθθ, gϕϕ, gtϕ} = Extract[metric, {{1, 1}, {2, 2}, {3, 3}, {4, 4}, {1, 4}}];
                                提取
    e0 = √(-(gϕϕ / (gtt gϕϕ - gtϕ gtϕ))) {1, 0, 0, -(gtϕ / gϕϕ)}; (*e_t*)
    e1 = {0, -(1/(√grr)), 0, 0}; (*e_r*)
    e2 = {0, 0, 1/(√gθθ), 0}; (*e_θ*)
    e3 = {0, 0, 0, -(1/(√gϕϕ))}; (*e_ϕ*)
    {e0, e1, e2, e3}];
(*get the initial direction of the light and the upper index 4-velocity*)
ClearAll[GetRayDirection];
清除全部
GetRayDirection[metric_, pos_, fov_, npix_][i_, j_] :=
```

```
  Module[{λdot, xscr, yscr, θx, ψx, vel, e0, e1, e2, e3, κ},
  模块
   {xscr, yscr} = (2 Tan[fov / 2] / npix) (# - (1 / 2) (npix + 1)) & /@ {i, j};
                     正切
   {θx, ψx} = WrapAngle[2 ArcTan[(1 / 2) (√(xscr^2 + yscr^2))], ArcTan[-yscr, -xscr]];
                          反正切                                  反正切
   vel = N@{Cos[θx], Sin[θx] Cos[ψx], Sin[θx] Sin[ψx]};
         … 余弦      正弦     余弦     正弦     正弦
   {e0, e1, e2, e3} = N@LocalTetrads[pos, metric];
                      数值运算
   κ = 1;
   λdot = -κ e0 + vel[[1]] e1 + vel[[2]] e2 + vel[[3]] e3;
   λdot];
(*calculate the upper index 4-velocity of the disk fluid*)
ClearAll[uDisk];
清除全部
uDisk[a_, r_, θ_, ϕ_] :=
  Module[{ut, ur, uϕ, Σ, Δ, guptt, guptϕ, gupϕϕ, gdowntt, gdowntϕ, gdownϕϕ, gdownrr},
  模块
   If[r > rISCO,
   如果
    (*outside the ISCO*)
    ut = N[Sqrt[(r^3 + a^2 (2 + r)) / (r (a^2 + (-2 + r) r))] /
           Sqrt[1 - (a^2 - 2 a Sqrt[r] + r^2)^2 / ((a^2 + (-2 + r) r) (a + r^(3/2))^2)]];
         数值运算
    ur = 0;
    uϕ = Sqrt[(r^3 + a^2 (2 + r)) / (r (a^2 + (-2 + r) r))] /
         ((a + r^(3/2)) Sqrt[1 - (a^2 - 2 a Sqrt[r] + r^2)^2 / ((a^2 + (-2 + r) r) (a + r^(3/2))^2)]);,
    (*inside the ISCO*)
    Σ = r^2;
    Δ = r^2 - 2 r + a^2;
    guptt = -1 - (2 r (r^2 + a^2)) / (Δ Σ);
    guptϕ = -2 a r / (Δ Σ);
    gupϕϕ = (Δ - a^2) / (Δ Σ);
    gdowntt = -(1 - (2 r) / Σ);
    gdowntϕ = -(1 / Σ) × 2 a r;
    gdownϕϕ = r^2 + a^2 + (1 / Σ) × 2 a^2 r ;
    gdownrr = Σ / Δ;
    ut = -eISCO guptt + lISCO guptϕ;
    uϕ = -eISCO guptϕ + lISCO gupϕϕ;
```

```
    ur = - Sqrt[(gdowntt ut^2 + 2 gdowntϕ ut uϕ + gdownϕϕ uϕ^2 + 1)/(-gdownrr)];];
   {ut, 0, ur, uϕ}];
(*Emission profile*)
J[r_] := Exp[-1/2 Log[r/horizon0]^2 - 2 Log[r/horizon0]];
         指数形式                          对数

(*numerical solve the eom of a single ray*)
ClearAll[TraceSingleRay];
清除全部
TraceSingleRay[a0_, pos_, fov_, npix_][i_, j_] :=
  Module[{λF = 3000, metric, horizon, λdot, eqns, E0, L0, idata,
  模块
     initialConditions, sol, frame, momentum, eom, intensity = 0,
     redshift = 1, redshift1 = 1, redshift2 = 1, imgorder = 0},
    horizon = N@(1 + √(1 - a0^2));
                数值运算
    metric = MetricDown[a0][pos];
    λdot = GetRayDirection[metric, pos, fov, npix][i, j];
    momentum = metric.λdot;
    {E0, L0} = {-momentum[[1]], momentum[[4]]};
    idata = Join[pos, {momentum[[2]], momentum[[3]]}];
            连接
    initialConditions = Thread[{t[0], r[0], θ[0], ϕ[0], pr[0], pθ[0]} == idata];
                        逐项作用
    eom = Thread[{t'[λ], r'[λ], θ'[λ], ϕ'[λ], pr'[λ], pθ'[λ]} == Deom];
          逐项作用
    eqns = N[Join[eom, initialConditions] /. {a → a0, EE → E0, LL → L0}];
           ·· 连接
    sol = NDSolve[eqns, {t, r, θ, ϕ, pr, pθ}, {λ, 0, λF}, Method → {"EventLocator",
          数值求解微分方程组                                      方法
        "Event" → {r[λ] - 1.01 horizon, r[λ] - 10000 horizon, θ[λ] - π/2}, "EventAction" :→
         {Throw[λF = λ, "StopIntegration"], Throw[λF = λ, "StopIntegration"],
          抛                               抛
          imgorder = imgorder + 1;
          If[1.01 horizon ≤ r[λ] ≤ 60 horizon,
          如果
           redshift = - 1/(uDisk[a0, r[λ], θ[λ], ϕ[λ]].{E0, -pr[λ], -pθ[λ], -L0});
           intensity = intensity + redshift^3 * J[r[λ]];
           If[imgorder == 1, redshift1 = redshift];
           如果
           If[imgorder == 2, redshift2 = redshift];]}
           如果
      }];
    {intensity, redshift1, redshift2, imgorder}];
```

```
(*Progress Indicator*)
ClearAll[MonitorParallelTable];
清除全部
MonitorParallelTable[expr_, npix_] := Module[{res, iterCount = npix^2, progress = 0},
                                      模块
 SetSharedVariable[progress];
 设置共享变量
 res = Monitor[ParallelTable[(progress++;
       监控     并行产生表格
        expr[j, i]), {i, 1, npix}, {j, 1, npix}],
  Column[{ToString@progress <> " of " <> ToString@iterCount,
  列      转换为字符串                      转换为字符串
        ProgressIndicator[progress, {0, iterCount}]}, Alignment → Center]];
        进度指示器                                       对齐          居中
 UnsetShared[progress];
 停止共享
   res];
(*trace all rays*)
ClearAll[TraceRay];
清除全部
TraceRay[a_, pos_, fov_, npix_] :=
  MonitorParallelTable[TraceSingleRay[a, pos, fov, npix], npix];
Print["-------------------------------Done
打印
   2----------------------------------------------"]
```

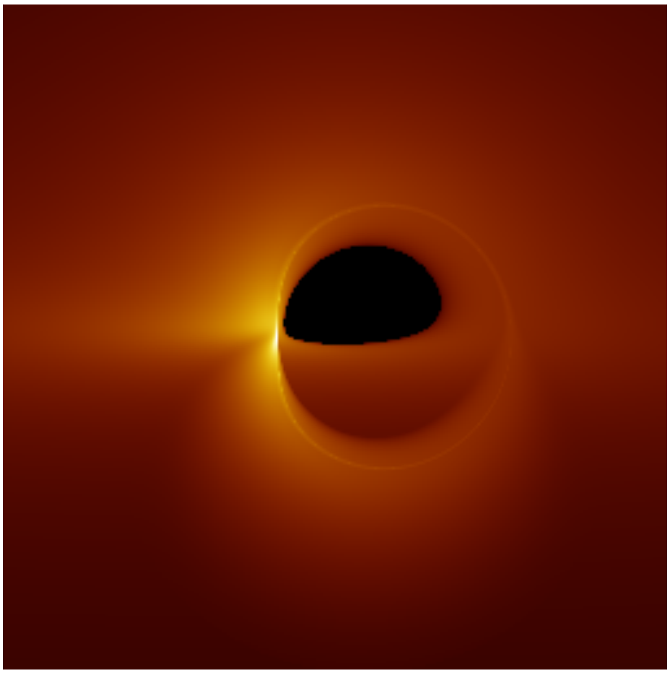
-------------------------------Done 2---

```
(*Begin here*)
  开始
ClearAll[result]
清除全部
a0 = 0.99;
horizon0 = 1 + Sqrt[1 - a0^2];
(*!!ATTENTION!!*)
(*!!These three numbers should be caculated by another code!!*)
rISCO = 1.4545; lISCO = 1.56836; eISCO = 0.73597;
pos = {0, 500, 80 π / 180, 0};
fov = 3 π / 180;
npix = 256; (*should be even*)
AbsoluteTiming[result = TraceRay[a0, pos, fov, npix];]
绝对时间
```

Out[∘]= {117.838, Null}

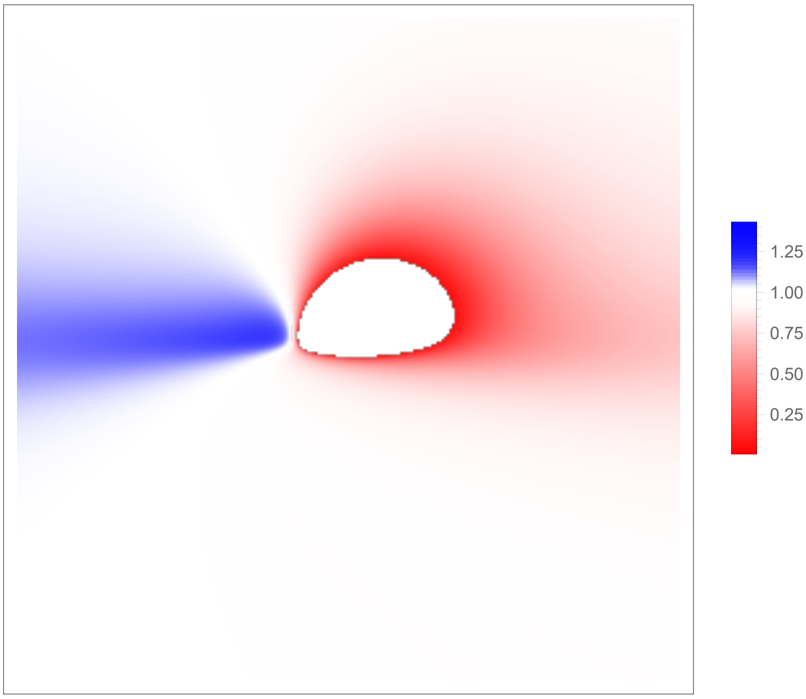
```
(*plot intesity*)
intenplot = Re@result[[All, All, 1]];
              实部        全部  全部
Imax = Max[intenplot];
       最大值
pltint =
 MatrixPlot[intenplot, DataReversed → {True, False}, AspectRatio → 1, Frame → False,
 矩阵图                  数据颠倒否         真     假           宽高比                边框    假
  ColorFunction → Function[a, RGBColor[(a / Imax)^(1/12), (a / Imax)^(1/4), (a / Imax)^2]],
  颜色函数              纯函数       RGB颜色
  ColorFunctionScaling → False, PlotLegends → None]
  颜色函数缩放            假       绘图的图例       无
(*GraphicsRow[{pltint,BarLegend[
  按行画出图形          条形图例
   {Function[a,RGBColor[(a/Imax)^(1/12),(a/Imax)^(1/4),(a/Imax)^2]],{0,Imax}}]}]*)
    纯函数       RGB颜色
```

Out[•]=



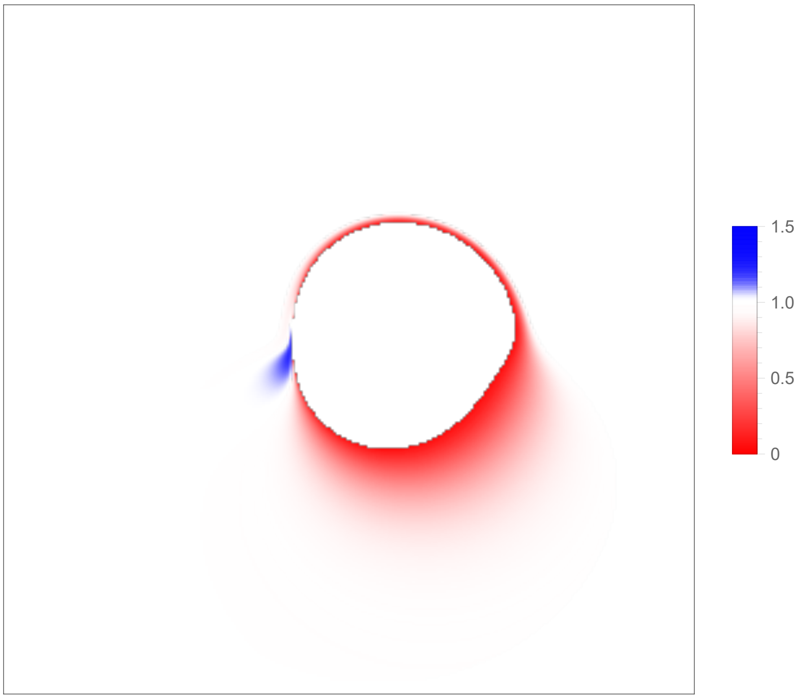
```
(*plot redshift for n=1*)
pltrs1 = ArrayPlot[Re@result⟦All, All, 2⟧,
              ⌊图示数组    ⌊实部         ⌊全部 ⌊全部
   DataReversed → {True, False}, PlotLegends → Automatic,
   ⌊数据颠倒否         ⌊真     ⌊假       ⌊绘图的图例       ⌊自动
   ColorFunction → Function[a, RGBColor[If[a < 1, 1, 3 * (1 - a)], If[a < 1, a, 3 * (1 - a)],
   ⌊颜色函数          ⌊纯函数       ⌊RGB颜色   ⌊如果                          ⌊如果
      If[a < 1, a, 1], 1 - 1 / (1 + (a - 1) ^2 / 0.05)]], ColorFunctionScaling → False]
      ⌊如果                                        ⌊颜色函数缩放             ⌊假
```

Out[∘]=



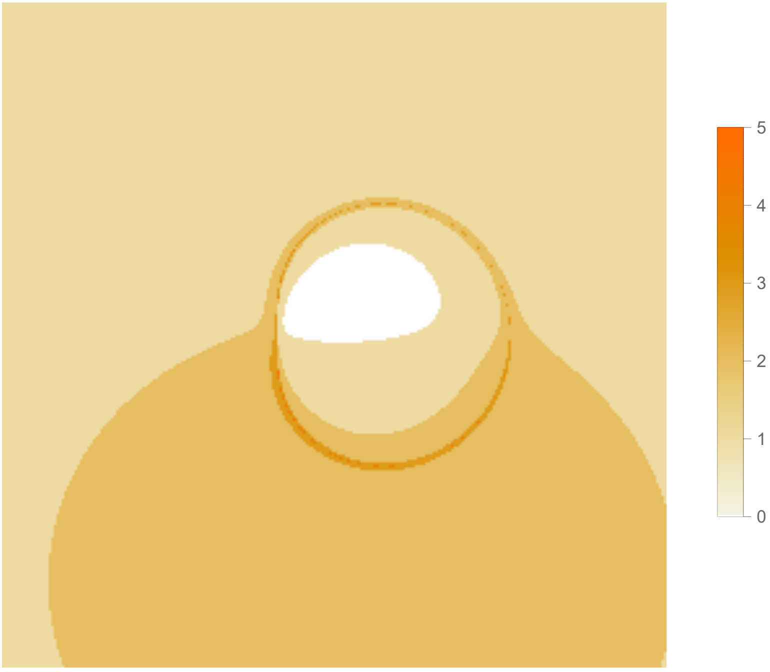
```
(*plot redshift for n=2*)
pltrs2 = ArrayPlot[Re@result[[All, All, 3]],
           图示数组     实部           全部  全部
  DataReversed → {True, False}, PlotLegends → Automatic,
  数据颠倒否         真     假         绘图的图例        自动
  ColorFunction →  Function[a, RGBColor[If[a < 1, 1, 3 * (1 - a)], If[a < 1, a, 3 * (1 - a)],
  颜色函数            纯函数         RGB颜色    如果                          如果
     If[a < 1, a, 1], 1 - 1 / (1 + (a - 1) ^2 / 0.05)]], ColorFunctionScaling → False]
     如果                                              颜色函数缩放           假
```

Out[ ]=



```
(*plot the maximum number of times that the ray crosses the equatorial plane*)
plttimes = MatrixPlot[result[[All, All, -1]], DataReversed → {True, False},
           |矩阵图            |全部 |全部       |数据颠倒否        |真     |假
  AspectRatio → 1, Frame → False, PlotLegends → Automatic]
  |宽高比            |边框     |假      |绘图的图例       |自动
```

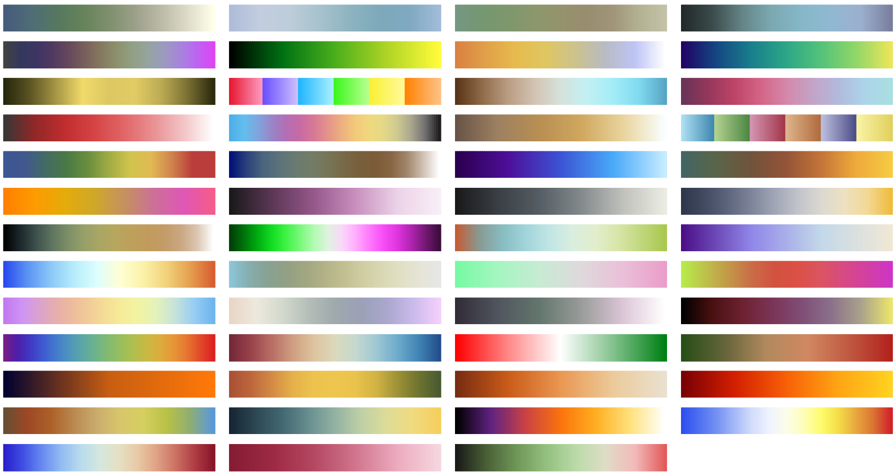
Out[ ]=



In[∘]:=
```
ColorData["Gradients"]
颜色数据
Grid[Partition[Show[ColorData[#, "Image"], ImageSize → 110] & /@
格子  划分      显示  颜色数据        图像       图像尺寸
    ColorData["Gradients"], 4, 4, 1, {}], Spacings → .5]
    颜色数据                           间隔
```

Out[∘]=
```
{AlpineColors, Aquamarine, ArmyColors, AtlanticColors, AuroraColors, AvocadoColors,
 BeachColors, BlueGreenYellow, BrassTones, BrightBands, BrownCyanTones,
 CandyColors, CherryTones, CMYKColors, CoffeeTones, DarkBands, DarkRainbow,
 DarkTerrain, DeepSeaColors, FallColors, FruitPunchColors, FuchsiaTones, GrayTones,
 GrayYellowTones, GreenBrownTerrain, GreenPinkTones, IslandColors, LakeColors,
 LightTemperatureMap, LightTerrain, MintColors, NeonColors, Pastel, PearlColors,
 PigeonTones, PlumColors, Rainbow, RedBlueTones, RedGreenSplit, RoseColors, RustTones,
 SandyTerrain, SiennaTones, SolarColors, SouthwestColors, StarryNightColors,
 SunsetColors, TemperatureMap, ThermometerColors, ValentineTones, WatermelonColors}
```

Out[∘]=



In[∘]:=
```
(*Export[FileNameJoin[{NotebookDirectory[],"rs_fov"<>ToString[fov*180/π]<>
  导出   文件名连接       当前笔记本的目录                转换为字符串
    "d_a"<>ToString[1000 a0]<>"_obs"<>ToString[pos[[3]]*180/π]<>"d.png"}],
           转换为字符串                 转换为字符串
  pltg,ImageSize→ 1600,ImageResolution→500]*)
       图像尺寸          图像分辨率
```