In[1065]:=

```
(* ========================================================================== *)
(* PART 1: 宇宙基底构建 (Computational Universe Generation) *)
(* ========================================================================== *)
ClearAll[];
SeedRandom[]; (* 你可以保留这个种子，或者删掉它进行全随机验证 *)

rccRigidStep[g_Graph] := Module[{
  allEdges, activeEdges, selectedPair, e1, e2, x, y, z, w,
  nextV, newActiveEdges, inertEdges
 },
 allEdges = EdgeList[g];
 activeEdges = Cases[allEdges, _UndirectedEdge];

 (* 蒙特卡洛采样加速 *)
 Block[{shuffled = RandomSample[activeEdges]},
  Do[
   e1 = shuffled[[i]];
   Do[
    e2 = shuffled[[j]];
    If[Length[Intersection[List @@ e1, List @@ e2]] == 1,
     selectedPair = {e1, e2};
     Goto["Found"];
    ],
    {j, i + 1, Min[i + 20, Length[shuffled]]}
   ],
   {i, Length[shuffled]}
  ]
 ];

 Label["Found"];
 If[Not[ValueQ[selectedPair]], Return[g]];

 {e1, e2} = selectedPair;
 y = Intersection[List @@ e1, List @@ e2][[1]];
 x = Complement[List @@ e1, {y}][[1]];
 z = Complement[List @@ e2, {y}][[1]];
 nextV = Max[VertexList[g]] + 1;
 w = nextV;

 newActiveEdges = {x ↔ z, x ↔ w, w ↔ z};
```

```
 inertEdges = {x → y, y → z};

 Graph[
  VertexList[g] ~Join~ {w},
  Union[Complement[allEdges, {e1, e2}], newActiveEdges, inertEdges]
 ]
];

(* 生成宇宙 *)
initG = CycleGraph[4];
steps = 1000; (* 步数建议保持 1000 以上以保证空间足够大 *)
Print["The vacuum environment for the double-slit
     experiment is being constructed (Steps: ", steps, ")..."];
universe = Nest[rccRigidStep, initG, steps];

(* ========================================================================= *)
(* PART 2: 量子演化 (Quantum Evolution) *)
(* ========================================================================= *)

(* 1. 安全提取拉普拉斯算子 *)
nodeList = VertexList[universe];
numNodes = Length[nodeList];
adj = AdjacencyMatrix[universe];
laplacian = N[DiagonalMatrix[Total[adj]] - adj];

(* 2. 设置波源 (动态寻找有效索引) *)
source1 = nodeList[[1]];
source2 = If[numNodes ≥ 4, nodeList[[4]], nodeList[[2]]];

(* 3. 初始化波函数 *)
psi = SparseArray[{
  FirstPosition[nodeList, source1][[1]] → 1.0,
  FirstPosition[nodeList, source2][[1]] → 1.0
}, {numNodes}] // Normal // N;
psi = psi / Norm[psi];

dt = 0.05;
evolutionSteps = 150;

Print["Quantum propagation in progress..."];
Do[
```

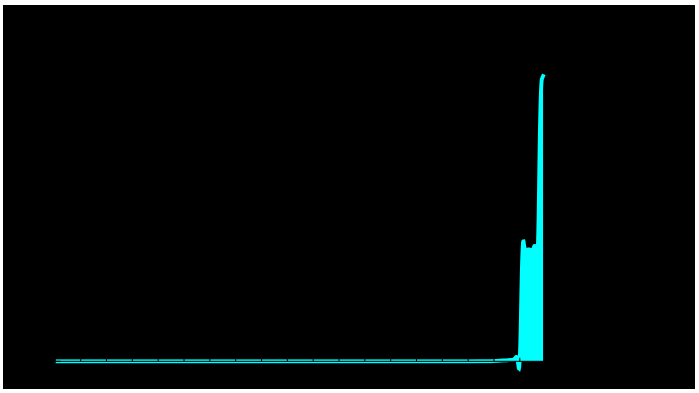
```
 psi = psi - I * dt * (laplacian . psi);
 psi = psi / Norm[psi];
, {evolutionSteps}];

(* ========================================================================== *)
(* PART 3: 干涉探测 (Interference Detection) - 【已优化：防太近】 *)
(* ========================================================================== *)
intensity = Abs[psi]^2;

(* 计算距离映射 *)
distMap = GraphDistance[universe, source1];

(* 寻找波前：找到平均概率密度最大的距离层 *)
validDists = Cases[distMap, _?NumberQ];
maxD = Max[validDists];

(* 【核心优化】：强制屏幕必须在宇宙深处，至少在前 30% 以后 *)
(* 这样能避开源点附近的混沌区，让波纹充分展开 *)
minSafeDist = Floor[maxD * 0.3];
minSafeDist = Max[minSafeDist, 6]; (* 并且绝对距离至少要大于 6 *)

layerAvg = Table[
 {d, Mean[Pick[intensity, distMap, d]]},
 {d, 2, maxD}
];

(* 筛选：只看安全距离之外的层，然后选其中强度最大的 *)
safeLayers = Select[layerAvg, First[#] > minSafeDist &];

If[Length[safeLayers] == 0,
 (* 备用方案：如果宇宙实在太小，就选最远层 *)
 bestDist = Last[SortBy[layerAvg, First]][[1]];
 Print["Warning: Universe too small, using furthest layer: ", bestDist];
,
 (* 正常方案：在安全区里找波峰 *)
 bestDist = First[MaximalBy[safeLayers, Last]][[1]];
];

Print["Optimized interference layer found at distance: ",
  bestDist, " (Safe zone > ", minSafeDist, ")"];
```

```
(* 提取并排序节点 *)
screenNodes = Select[nodeList, GraphDistance[universe, source1, #] == bestDist &];

If[Length[screenNodes] < 5,
 Print["Warning: Space too small for interference. Increase steps in Part 1."];
,
 (* 按到 source2 的距离排序以展开波纹 *)
 sortedIntensities = SortBy[
  Table[{GraphDistance[universe, source2, node],
       intensity[[FirstPosition[nodeList, node][[1]]]]}, {node, screenNodes}],
  First
 ];

 (* 绘图 *)
 Print[ListLinePlot[sortedIntensities[[All, 2]],
  PlotRange → All,
  PlotStyle → {Thick, Cyan},
  Filling → Axis,
  FillingStyle → Opacity[0.3, Cyan],
  InterpolationOrder → 2,
  PlotLabel → "Interference Pattern at Distance " <> ToString[bestDist],
  AxesLabel → {"Screen Position", "Intensity"},
  Background → Black,
  BaseStyle → {FontColor → White}
 ]];
];

(* ================================================================== *)
(* PART 4: 干涉验证与观察者效应 (Interference Verification) *)
(* ================================================================== *)

Print[];

(* 模拟单缝/测量坍缩 *)
psiCollapsed = ConstantArray[0.0 + 0.0 I, numNodes];
psiCollapsed[[source1]] = 1.0;

Do[
 psiCollapsed = psiCollapsed - I * dt * (laplacian . psiCollapsed);
 psiCollapsed = psiCollapsed / Norm[psiCollapsed];
```

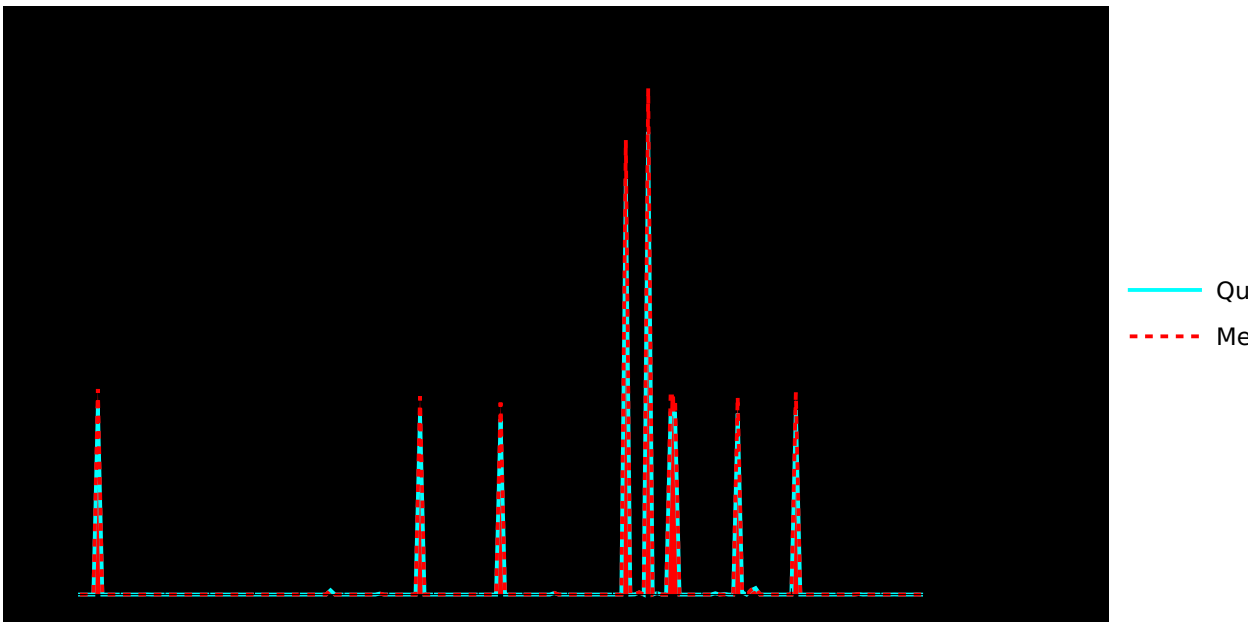
```
 , {t, 1, evolutionSteps}
];

intensityCollapsed = Abs[psiCollapsed]^2;

(* 再次应用安全距离逻辑，确保对比图也是在有效区域 *)
(* 直接复用 Part 3 找到的最佳距离 bestDist，确保在同一个屏幕上对比 *)
(* 这样对比才公平 *)
targetNodes = Select[nodeList, GraphDistance[universe, source1, #] == bestDist &];

(* 只有当该层有节点时才绘图 *)
If[Length[targetNodes] > 0,
 interferePattern = intensity[[FirstPosition[nodeList, #][[1]] & /@ targetNodes]];
 collapsePattern = intensityCollapsed[[FirstPosition[nodeList, #][[1]] & /@ targetNodes]];

 (* 为了画图平滑，同样按照到 source2 的距离排序 *)
 sortOrder = Ordering[GraphDistance[universe, source2, #] & /@ targetNodes];
 interferePattern = interferePattern[[sortOrder]];
 collapsePattern = collapsePattern[[sortOrder]];

 interferenceTitle =  <> ToString[bestDist];

 vizComparison = ListLinePlot[
  {interferePattern, collapsePattern},
  PlotRange → All,
  PlotStyle → {{Thick, Cyan}, {Thick, Red, Dashed}},
  Filling → Axis,
  PlotLegends → {, },
  PlotLabel → interferenceTitle,
  AxesLabel → {, },
  Background → Black,
  BaseStyle → {FontColor → White},
  ImageSize → Large
 ];

 Print[vizComparison];
 Print[];
,
 Print[];
];
The vacuum environment for the double-slit experiment is being constructed (Steps:
 1000)...
```

Quantum propagation in progress...

Optimized interference layer found at distance: 7 (Safe zone > 6)



Simulating Observer Effect (Measurement at Slits)...



If the Cyan line shows more 'peaks'
(M-shape) than the Red line, interference is confirmed.