Speciální grafy

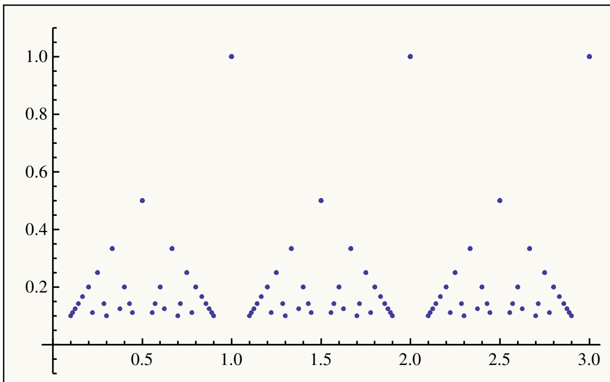
Nyní se podíváme na grafické zachycení specifických funkcí, které nelze v programu *Mathematica* vykreslit jednoduchým způsobem, ukázaným v předchozí části. Jsou to funkce, které jsou buď definovány po nekonečně mnoha částech nebo funkce, jejichž definiční obor není složen z intervalů, např. Riemannova funkce, Cantorova funkce nebo spojité dunkce nemající derivace v žádných bodech (viz kapitolu o derivacích).
Podívejme se nejdříve na Riemannovu funkci, která má hodnotu 0 v iracionálních bodech a v 0, hodnotu 1/q v racionálních bodech p/q, kde p,q jsou nesoudělná celá čísla a q>0.

Riemannova funkce

U funkcí typu Riemannovy funkce je možné pro vizualizaci nakreslit jen konečně mnoho bodů grafu a to tolik, aby z nich bylo možné rozpoznat nějaké vlastnosti funkce. U Riemannovy funkce není třeba vykreslovat nulové hodnoty v iracionálních bodech. Vezmeme některá racionální čísla na intervalu [0,3] a v nich nakreslíme hodnoty. Volitelné číslo k na začátku udává počet hladin v grafu.
Uvedené příkazy snadno rozluštíte (je vhodné si vyzkoušet, co dostanete, když např. vynecháte příkaz Flatten nebo místo Table napíšete List. Zajímavá je definice funkce f2 (rozeberte si ji). Uvedená demonstrace Wolframu je obtížnější.

```
k = 10;
```

```
t1 = Flatten[Table[{n, m}, {m, 1, k}, {n, 1, 3 m}], 1];
f1[{a_, b_}] := a / b;  f2[{a_, b_}] := 1 / {a / b // Denominator};
tx1 = Map[f1, t1]; tx2 = Flatten[List[Transpose[Map[f2, t1]]]];
ListPlot[Table[{tx1[[i]], tx2[[i]]}, {i, 1, Length[tx1]}],
 PlotRange → {-.1, 1.1}]
```

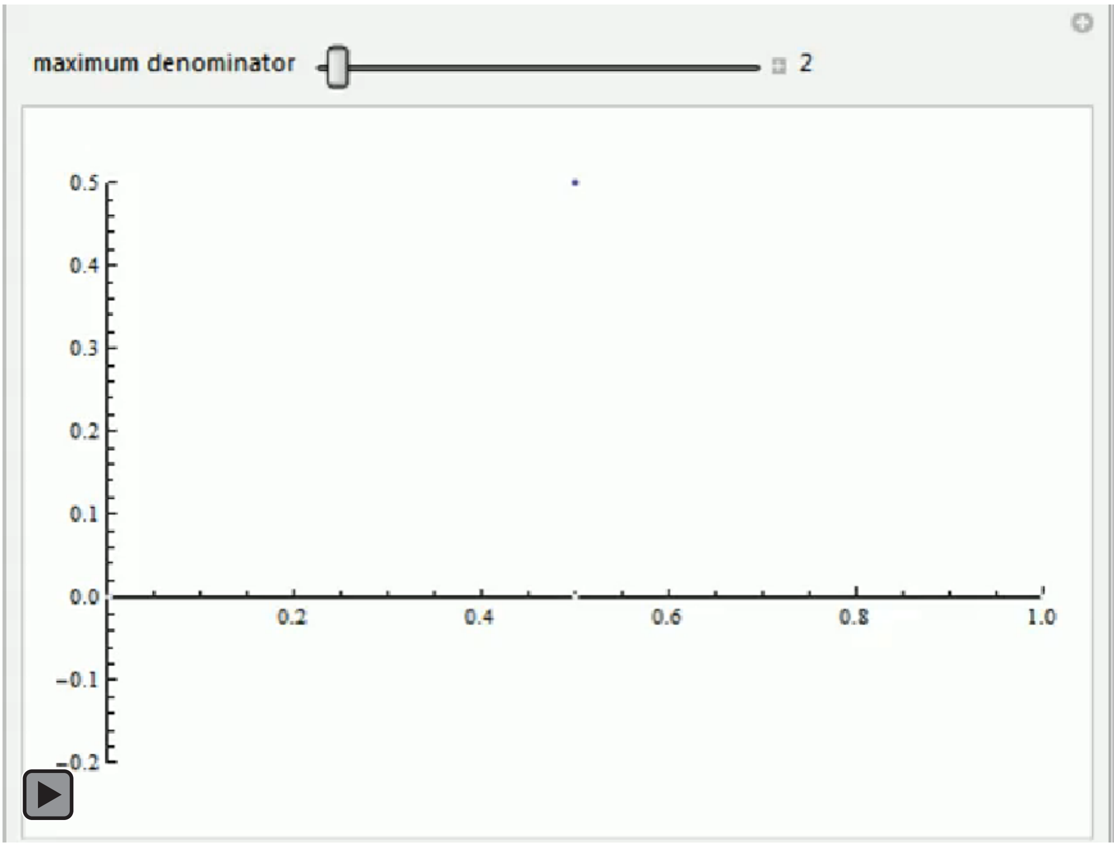


Chytřejší postup je uveden v demonstracích Wolframu (mírně ho upravíme podle našich potřeb):

```
Riemann[n_] := Sort[Join[{{0, 0}, {1, 0}},
   Join @@ Table[Thread[{Select[Range[i], GCD[i, #] == 1 &], 1}] /
      i, {i, 2, n}]]]
```

In[22]:=

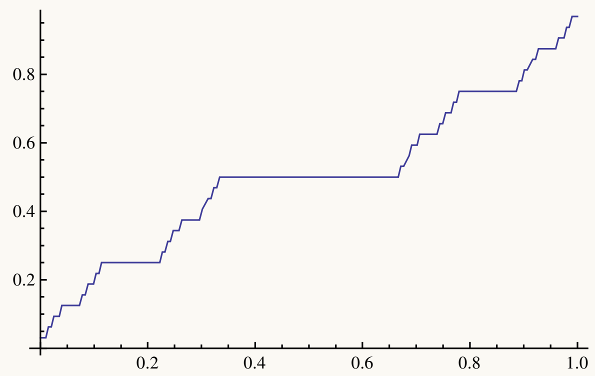
```
a =
 Manipulate[ListPlot[Riemann[n],
   PlotRange → {{0, 1}, {-.2, 1 / 2}}, ImageSize → {450, 300},
   Axes → True, AxesOrigin → {0, 0},
   Epilog → {Line[{{0, 0}, {1, 0}}], White,
     Point[{First@#, 0} & /@ Riemann[n]]}],
  {{n, 2, "maximum denominator"}, 2, 80, 1,
   Appearance → "Labeled"},
  SaveDefinitions → True]
```



Cantorova funkce

Trochu jinak se kreslí Cantorova funkce, která je konstantní na mnoha intervalech. Je definována na [0,1], neklesající a v bodě s trojkovým rozvojem {2 a_n} (každý trojkový rozvoj se dá napsat jen pomocí nul a dvojek) má hodnotu {a_n}. Zkuste následující popis funkce cantor rozebrat. V příkazu grafiky si ověřte význam volby Speed.

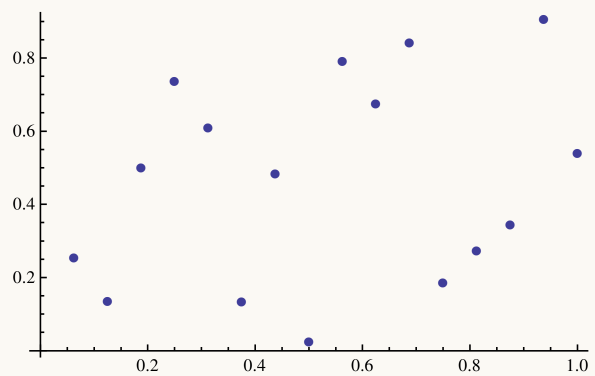
```
Clear[a, b, x, troj, konec, cantor];
troj[x_] :=
 RecurrenceTable[{a[n + 1] == Floor[3 b[n]],
   b[n + 1] == 3 b[n] - Floor[3 b[n]], a[1] == 0, b[1] == x},
  {a, b}, {n, 1, 5}];
t[x_] := Transpose[troj[x]][[1]];
konec[x_] := If[Position[t[x], 1.] == {}, 6,
   Position[t[x], 1., 1, 1][[1]][[1]]];
cantor[x_] := Sum[t[x][[k]] / 2^(k), {k, 1, konec[x] - 1}] +
   1 / 2^(konec[x] - 1);
Plot[cantor[x], {x, 0, 1}, PerformanceGoal → "Speed"]
```



Funkce s konečnými definičními obory

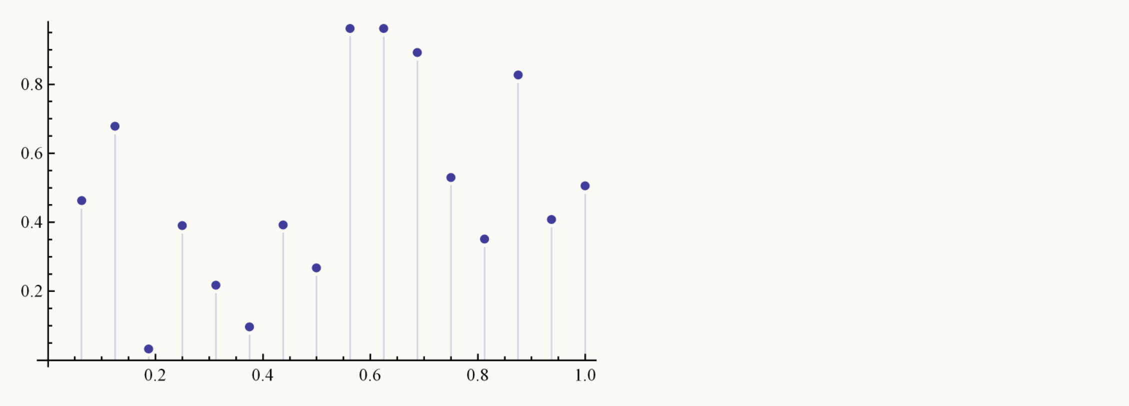
Funkce definované na konečných množinách mají graf složený z jednotlivých bodů, které spolu nejsou spojeny (někdy se ale může hodit je spolu spojit). Pro tento případ se používá často příkaz LisPlot.
Mějme např. nakreslit graf funkce definované na bodech $k/2^n$, k=1,...,8, n=0,...,3, s náhodnými hodnotami .

```
domain =
  Union[Flatten[Table[k / 2^n, {n, 0, 4}, {k, 1, 2^n}]]];
ListPlot[Table[{domain[[k]], Random[]},
  {k, 1, Length[domain]}], PlotMarkers → Automatic]
```



ListPlot má opět mnoho možností voleb, některé společné s Plot, některé jiné. Pro zlepšení orientace v grafu je vhodné upravit graf následovně.

```
ListPlot[Table[{domain[[k]], Random[]},
   {k, 1, Length[domain]}], PlotMarkers → Automatic,
  Filling → Axis]
```



Někdy býví vhodné hodnoty spojit.

```
ListPlot[Table[{domain[[k]], Random[]},
   {k, 1, Length[domain]}], PlotMarkers → Automatic,
  Joined -> True]
```