Solve Delay Differential Equations - Wolfram Mathematica

Решение дифференциальных уравнений с запаздывающим аргументом

Вы можете использовать стандартную функцию для решения дифференциальных уравнений, NDSolve , для численного решения дифференциальных уравнений с запаздывающим аргументом (ДУЗА) с фиксированными запаздываниями. При этом, NDSolve выдает результат в виде интерполирующей функции, которая затем может быть легко использована совместно с другими функциями.

Возьмем дифференциальное уравнение первого порядка с запаздывающим аргументом, с задержкой 1 и функцией предыстории t^2. Воспользуемся функцией NDSolve для решения уравнения и сохраним решение под именем sol1. Первым аргументом функции NDSolve является ДУЗА, вторым аргументом является переменная, для которой находится решение, а третьим аргументом является диапазон значений переменной:

In[1]:=

```
sol1 = NDSolve[{x'[t] == x[t - 1] (1 - x[t]), x[t /; t ≤ 0] == t^2}, x, {t, -
```

Out[1]=

```
{{x → InterpolatingFunction[{{-2., 2.}}, <>]}}
```

При помощи функции [Plot] построим график решения x[t] и его первой производной x'[t
] .
Для того, чтобы отобразить их графики разным цветом, во& #1089;пользуемся функцией
[Evaluate]
, "обернув" ее вокруг списка из функции и ее производной. Понадобится, также, загрузить модуль легенд (Plot Legends Package), если необходимо добавить пояснения к графику:

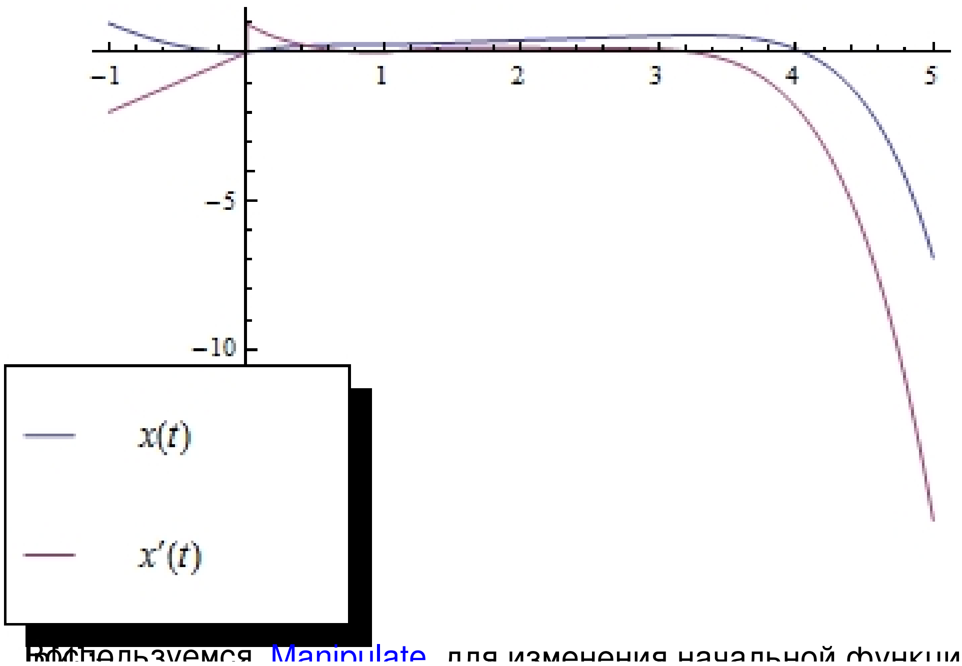
In[2]:=

```
Needs["PlotLegends`"]
```

In[3]:=

```
Plot[Evaluate[{x[t], x'[t]} /. First[sol1]], {t, -1, 5},
 PlotRange → All, PlotLegend → {Style[x[t], 14], Style[ x'[t], 14]}]
```

Out[3]=



Воспользуемся Manipulate для изменения начальной функции:

```
Manipulate[
 Module[
  {sol = NDSolve[{x'[t] == x[t - 1] (1 - x[t]), x[t /; t ≤ 0] == ϕ}, x, {t,
   Plot[Evaluate[x[t] /. First[sol]], {t, -2, 2}]],
 {ϕ, {0.1, t^2, Exp[t], Cos[t], Sin[t]}}]
```

Out[4]=

Решим ДУЗА первого порядка с двумя фиксированными запаздываниями, ? и 1, и функцией

```
sol2 =
 NDSolve[{x'[t] == x[t] (x[t - π] - x'[t - 1]), x[t /; t ≤ 0] == Cos[t]}, x,
```

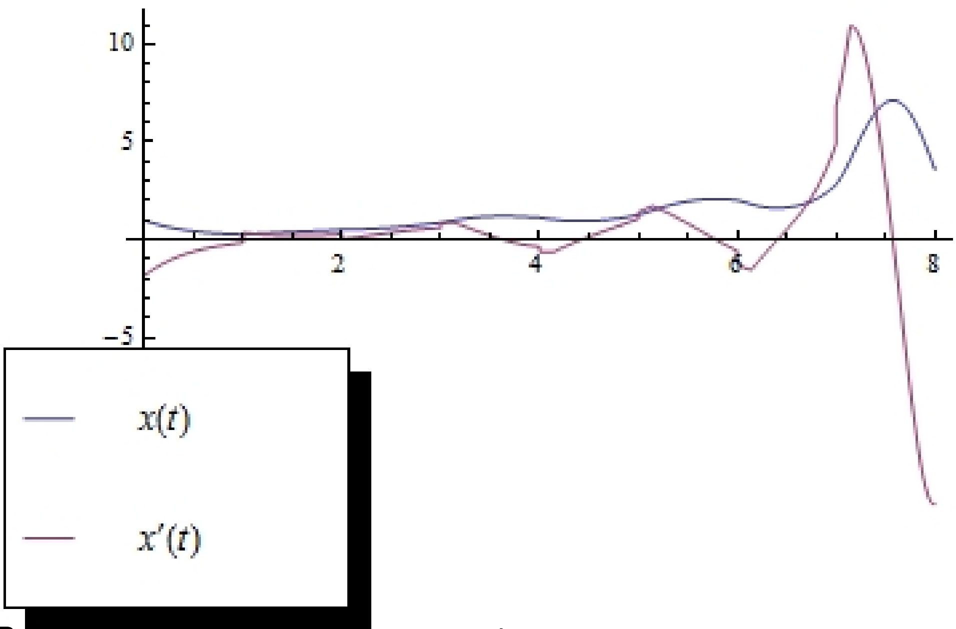
Out[5]=

```
{{x → InterpolatingFunction[{{0., 8.}}, <>]}}
```

При помощи функции Plot построим график решения x(t) и его первой производной x'[t

```
Plot[Evaluate[{x[t], x'[t]} /. First[sol2]], {t, 0, 8},
 PlotRange → All, PlotLegend → {Style[x[t], 14], Style[ x'[t], 14]}]
```

Out[6]=

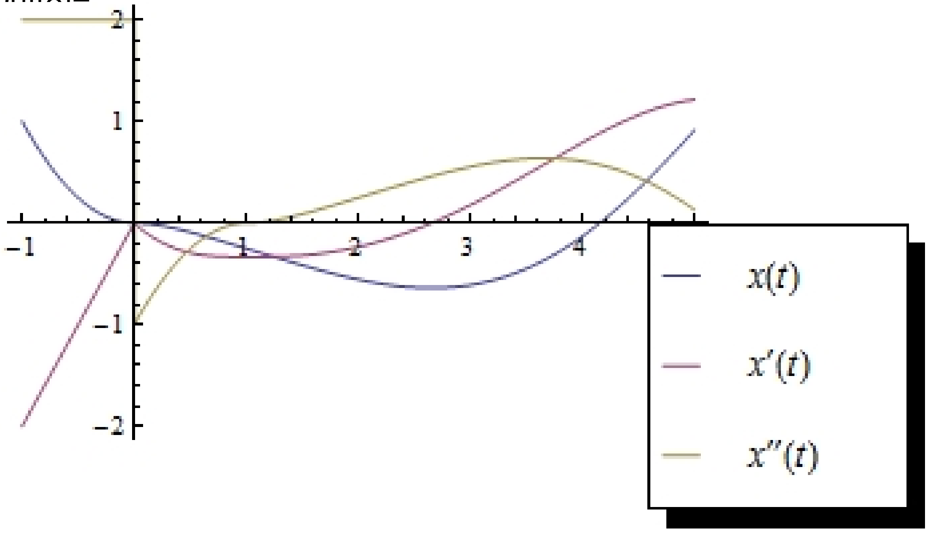


Решим ДУЗА второго порядка с фиксированным запаздыванием 1 и функцией предыстории t^2 : In[7]:=

```
sol3 = NDSolve[{x''[t] + x[t - 1] == 0, x[t /; t ≤ 0] == t^2}, x, {t, -1, 5
Out[7]=
{{x → InterpolatingFunction[{{-1., 5.}}, <>]}}
```

При помощи функции Plot построим график решения x[t] и его первых ... x''[t]. Для

```
Plot[Evaluate[{x[t], x'[t], x''[t]} /. First[sol3]], {t, -1, 5}, Plot
  PlotLegend → {Style[x[t], 14], Style[ x'[t], 14], Style[x''[t], 14]}
  LegendPosition → {0.8, -0.8}]
Out[8]=
```



Для более подробного ознакомления с уравнениями с запаздывающим аргументом, обратитесь к разделу Delay Differential Equations