Solitons in the Korteweg-de Vries Equation (KdV Equation)

In[15]:=
```
Clear["Global`*"]
```

Introduction

The Korteweg-de Vries Equation (KdV equation) describes the theory of water waves in shallow channels, such as a canal. It is a non-linear equation which exhibits special solutions, known as *solitons,* which are stable and do not disperse with time. Furthermore there are solutions with more than one soliton which can move towards each other, interact and then emerge at the same speed with no change in shape (but with a time "lag" or "delay").

The KdV equation is

$$\frac{\partial u}{\partial t} = 6\,u\,\frac{\partial u}{\partial x} - \frac{\partial^3 u}{\partial x^3}$$

Because of the u ∂u/∂x term the equation is non-linear (this term increases four times if u is doubled).

One soliton solution

The simplest soliton solution is

$$u(x, t) = -2\,\mathrm{sech}^2(x - 4\,t),$$

which is a trough of depth 2 traveling to the right with speed 4 and not changing its shape.

Let us verify that it does satisfy the equation:

In[16]:=
```
uexact[x_, t_] = -2 Sech[x - 4 t]^2
```

Out[16]=
```
-2 Sech[4 t - x]^2
```

In[17]:=
```
D[uexact[x, t], t] ==
  6 uexact[x, t] D[uexact[x, t], x] - D[uexact[x, t], {x, 3}] // Simplify
```

Out[17]=
```
True
```

Mathematica returns **True**, indicating that equation is satisfied.

Mathematica function **NDSolve** can solve partial differential equations in two (but not more than two) variables, such as x and t. However, it tends to be very slow and require a lot of memory. Nonetheless, if we put in the soliton at the initial time, it correctly propagates the soliton in time:

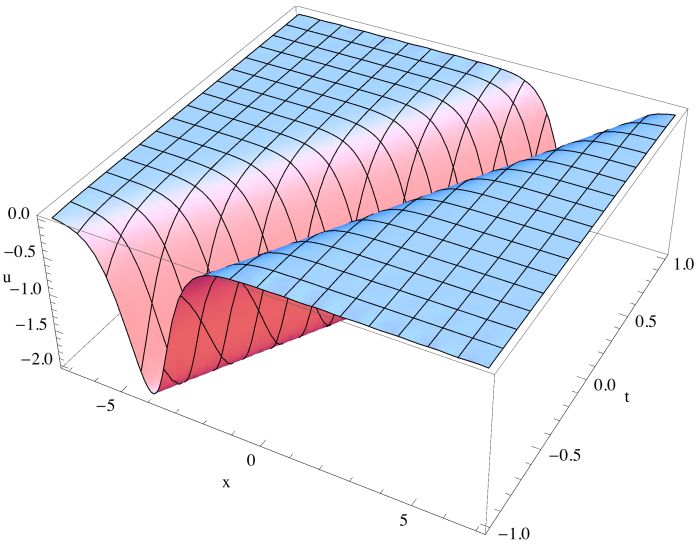
In[44]:=
```
xmin = -8; xmax = 8;
sol = NDSolve[ {D[u[x, t], t] == 6 u[x, t] D[u[x, t], x] - D[u[x, t], {x, 3}],
   u[x, 0] == -2 Sech[x]^2, u[xmin, t] == u[xmax, t] }, u, {x, xmin, xmax}, {t, -1, 1} ]
```

Out[44]=
```
{{u → InterpolatingFunction[{{-8., 8.}, {-1., 1.}}, <>]}}
```

Plotting the solution shows the trough propagating to the right.

In[19]:=
```
Plot3D[u[x, t] /. Flatten[sol], {x, -7, 7}, {t, -1, 1},
 PlotPoints → 50, PlotRange → All, AxesLabel → {"x", "t", "u"}]
```

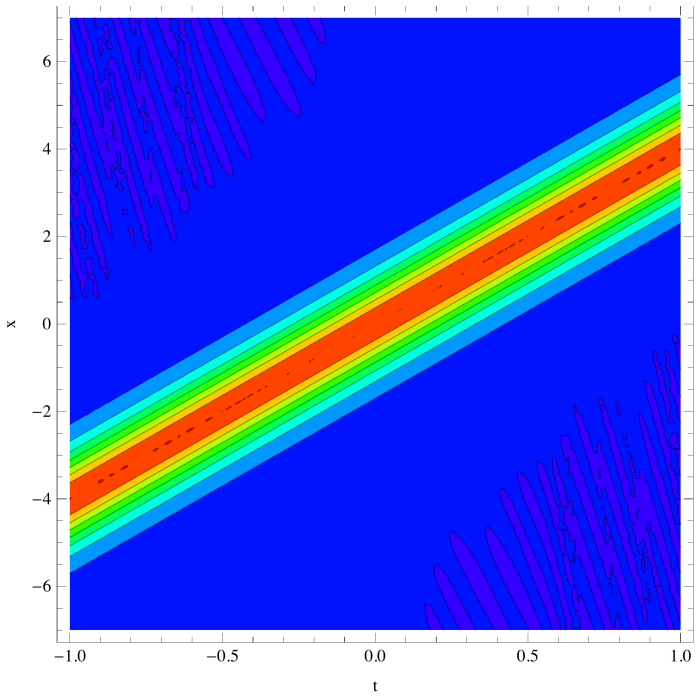
Out[19]=



A contour plot can also be useful:

In[20]:=
```
ContourPlot[u[x, t] /. Flatten[sol],
 {t, -1, 1}, {x, -7, 7}, ColorFunction → (Hue[0.7 #] &),
 PlotPoints → 50, PlotRange → All, FrameLabel → {"t", "x"}]
```

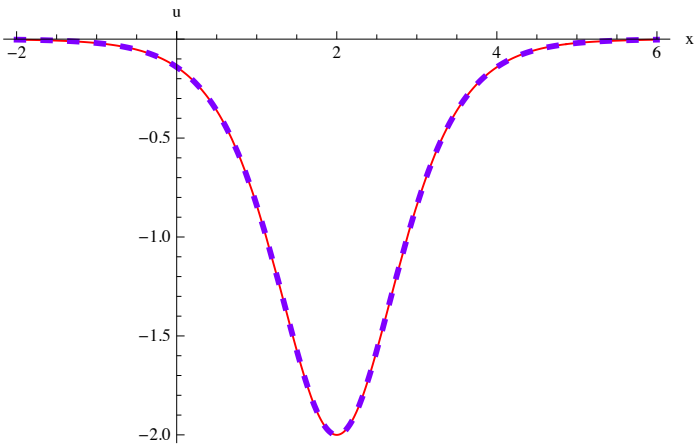
Out[20]=



To verify that the numerical solution is the soliton, we plot both for a particular value of t (t = 0.5 here):

In[21]:=
```
Plot[ {u[x, 0.5] /. Flatten[sol], -2 Sech[x - 2]^2}, {x, -2, 6}, PlotStyle →
  {{Hue[0], AbsoluteThickness[1]},
   {Hue[0.75], Dashing[{0.01, 0.03}], AbsoluteThickness[3]}}, AxesLabel → {"x", "u"}]
```

Out[21]=



We see that the two agree very well.

In fact there is a whole family of 1-soliton solutions parametrized by the depth of the trough. These are

$$u(x, t) = -u_{max}\,\mathrm{sech}^2\left[\sqrt{\frac{u_{max}}{2}}\,(x - 2\,u_{max}\,t)\right],$$

so the deeper the trough the faster the soliton moves and the narrower it is. To be precise

$$v = 2\,u_{max}\,.$$

We verify that this does satisfy the KdV equation:

In[22]:=
```
Clear[umax]
```

In[23]:=
```
uexact[x_, t_] = -umax Sech[Sqrt[umax / 2] (x - 2 umax t)]^2
```

Out[23]=
$$-\mathrm{umax}\,\mathrm{Sech}\left[\frac{\sqrt{\mathrm{umax}}\,(-2\,t\,\mathrm{umax} + x)}{\sqrt{2}}\right]^2$$

In[24]:=
```
D[uexact[x, t], t] ==
  6 uexact[x, t] D[uexact[x, t], x] - D[uexact[x, t], {x, 3}] // Simplify
```

Out[24]=
```
True
```

■ Two soliton solution

The theory for solutions with more than one soliton is complicated and we will not discuss it, but rather just *display* a two-soliton solution, verify that it is indeed a solution, and look at its properties. I failed to get *Mathematica* to integrate the solution forward in time. Specifying adequate resolution and number of time steps, my computer ran out of memory.

The theory states that an initial state

$$u(x, 0) = -n(n+1)\,\mathrm{sech}^2(x),$$

results in n solitons that propagate with different velocities. The solution for n = 2 is

$$u(x, t) = -12\,(3 + 4\cosh(2x - 8t) + \cosh(4x - 64t)) \,/\, [3\cosh(x - 28t) + \cosh(3x - 36t)]^2$$

It is not immediately evident that this satisfies the equation, but *Mathematica* confirms that it does:

In[25]:=
```
uexact[x_, t_] =
 -12 (3 + 4 Cosh[2 x - 8 t] + Cosh[4 x - 64 t]) / (3 Cosh[x - 28 t] + Cosh[3 x - 36 t])^2
```

Out[25]=
$$-\left(12\,(3+\text{Cosh}[64\,t-4\,x]+4\,\text{Cosh}[8\,t-2\,x])\right)\Big/\left(\text{Cosh}[36\,t-3\,x]+3\,\text{Cosh}[28\,t-x]\right)^2$$

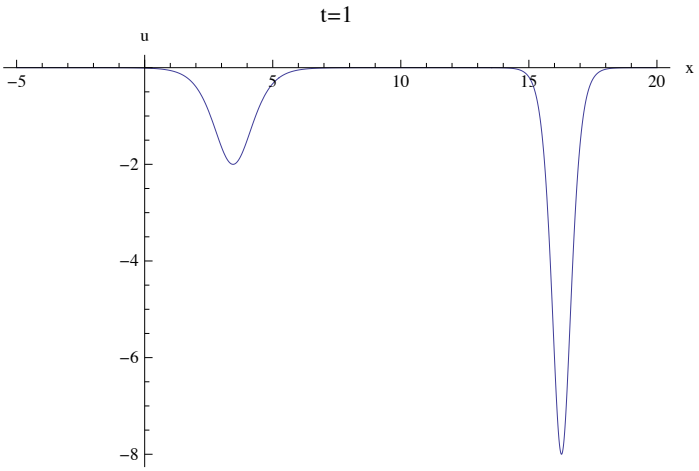
In[26]:=
```
D[uexact[x, t], t] ==
  6 uexact[x, t] D[uexact[x, t], x] - D[uexact[x, t], {x, 3}] // Simplify
```

Out[26]=
```
True
```

Next we plot the solution at time t = 1:

In[27]:=
```
Plot[uexact[x, 1], {x, -5, 20},
 PlotRange → All, PlotLabel → "t=1", AxesLabel → {"x", "u"}]
```

Out[27]=



We see a trough of depth 8 and a trough of depth 2. To determine the speeds of these troughs we locate the minima of the function at two different times, t=2 and 3,

In[28]:=
```
FindMinimum[uexact[x, 2], {x, 10}]
```

Out[28]=
```
{-2., {x → 7.45069}}
```

In[29]:=
```
FindMinimum[uexact[x, 3], {x, 10}]
```

Out[29]=
```
{-2., {x → 11.4507}}
```

In[30]:=
```
FindMinimum[uexact[x, 2], {x, 30}]
```

Out[30]=
```
{-8., {x → 32.2747}}
```

In[31]:=
```
FindMinimum[uexact[x, 3], {x, 50}]
```

Out[31]=
```
{-8., {x → 48.2747}}
```

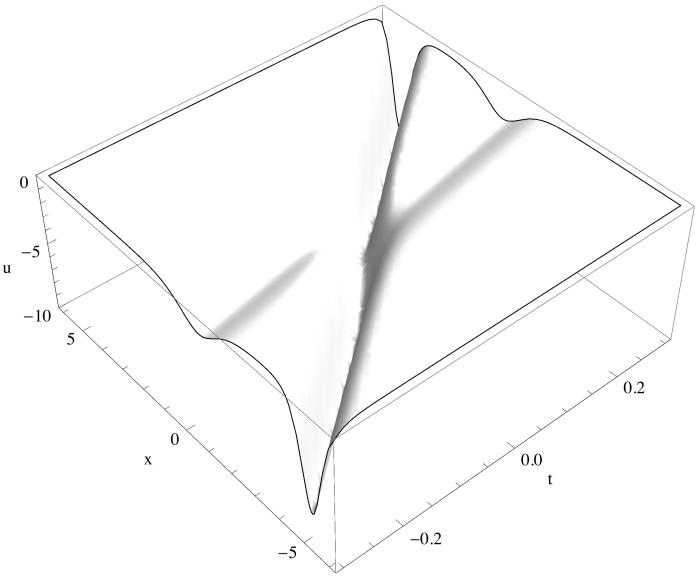
from which we deduce that the trough of depth 8 travels with speed 16 and the trough of depth 2 travels with speed 4. Thus we have created two solitons of the type that we discussed in the previous section.

Note, however, that **there is NO linear superposition** (because the equation is non-linear), so the 2-soliton solution is *not* the sum of the two individual solitons in the region where they overlap, as one can see from the explicit solutions.

Let's now see these two solitons interact in the vicninity of t = 0. We do a 3D plot, and present it twice with different options (one may look better printed on black and white; the other better on the screen):

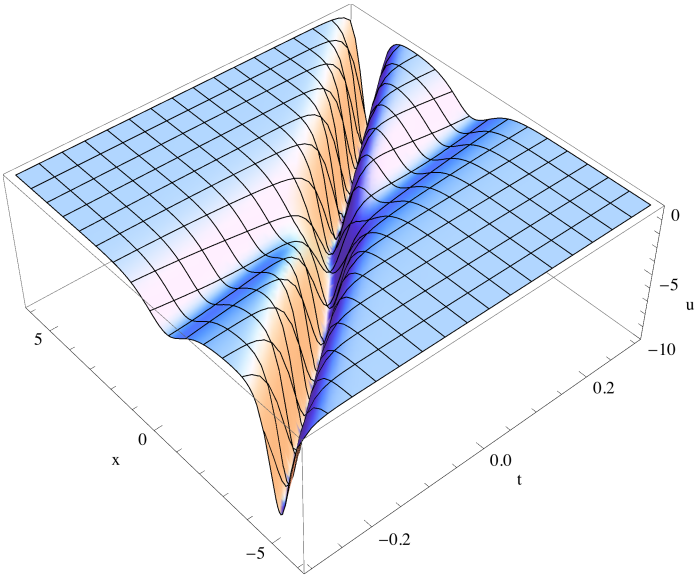
In[32]:=
```
Plot3D[uexact[x, t], {t, -0.3, 0.3}, {x, -6, 6}, PlotPoints → 50, PlotRange → {-10, 0},
 MeshStyle → Thickness[0], AxesEdge → {Automatic, Automatic, {-1, 1}},
 AxesLabel → {"t", "x", "u"}, ViewPoint → {-1.78, -2.06, 2.5}, ColorFunction → (White &)]
```

Out[32]=



In[33]:=
```
Plot3D[uexact[x, t] , {t, -0.3, 0.3}, {x, -6, 6},  PlotPoints → 50,
 PlotRange → {-10, 0}, AxesLabel → {"t", "x", "u"}, ViewPoint → {-1.78, -2.06, 2.5}]
```

Out[33]=

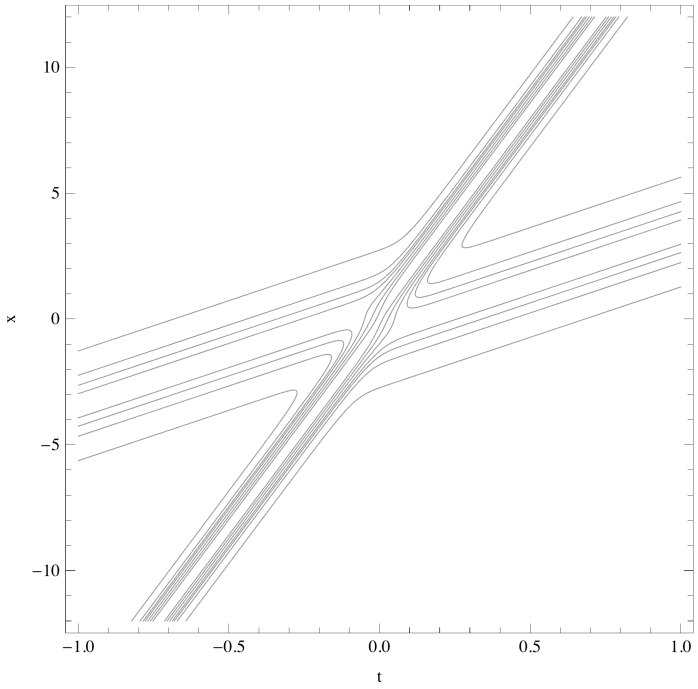


At negative times, the deeper soliton, which moves faster, approaches the shallower one. At t = 0 they combine to form the original solution, $u(x, 0) = -6\,\mathrm{sech}^2(x)$, (a single trough of depth 6) and, after the encounter, the deeper soliton has overtaken the shallower one and both resume their original shape and speed. However, as a result of the interaction, the shallower, slower soliton experiences is delayed and the deeper, faster soliton is advanced relative to their positions in the absence of the interaction between them.

The advance and delay are also easily seen in a contour plot (which again we present in two versions)

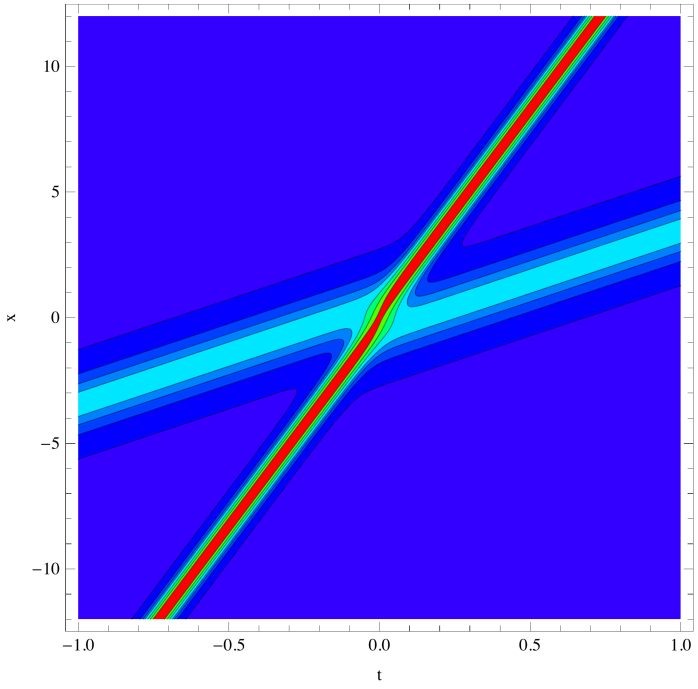
In[34]:=
```
ContourPlot[uexact[x, t] , {t, -1, 1}, {x, -12, 12},
 PlotPoints → 100, FrameLabel → {"t", "x"}, ContourShading → False,
 PlotRange → All, Contours → {-0.1, -0.6, -1.1, -1.6, -2.6, -4.0, -5.6}]
```

Out[34]=



In[35]:=
```
ContourPlot[uexact[x, t] , {t, -1, 1}, {x, -12, 12},
 PlotPoints → 100, FrameLabel → {"t", "x"}, ColorFunction → (Hue[0.7 #] &),
 PlotRange → All, Contours → {-0.1, -0.6, -1.1, -1.6, -2.6, -4.0, -5.6}]
```

Out[35]=



■ Other solutions

Now suppose that the initial condition is such that is does **not** just produce one or more solitons. We will take

$u(x, 0) = -4\,\mathrm{sech}^2(x)$,

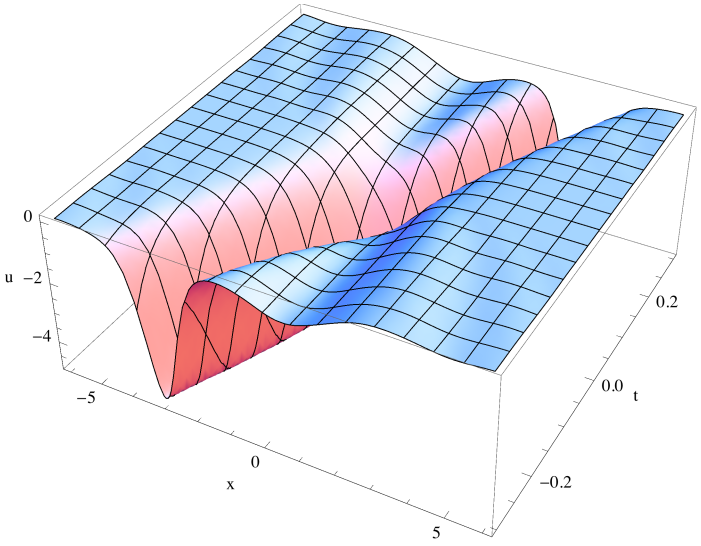
In[36]:=
```
xmin = -6; xmax = 6; sol = NDSolve[
  {D[u[x, t], t] == 6 u[x, t] D[u[x, t], x] - D[u[x, t], {x, 3}], u[x, 0] == -4 Sech[x]^2,
   u[xmin, t] == u[xmax, t]}, u, {x, xmin, xmax}, {t, -0.35, 0.35}]
```

Out[36]=
```
{{u → InterpolatingFunction[{{-6., 6.}, {-0.35, 0.35}}, <>]}}
```

Ignoring the warning messages, we plot the solution:

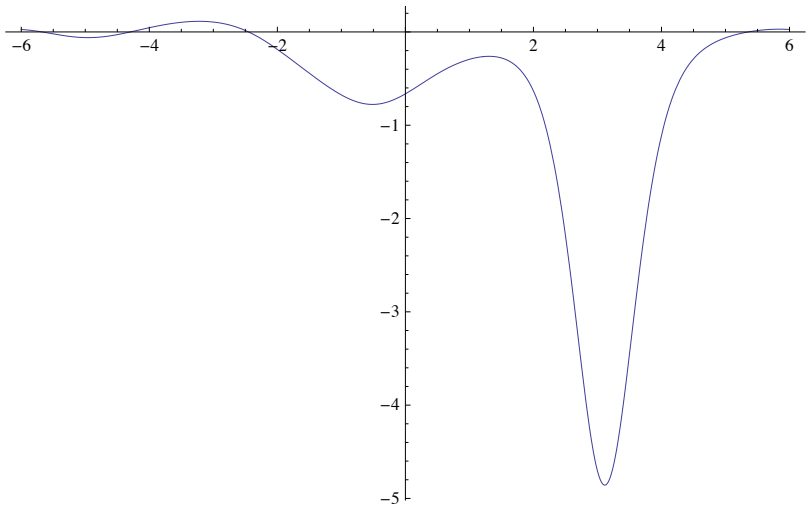
In[37]:=
```
Plot3D[u[x, t] /. Flatten[sol], {x, -6, 6}, {t, -0.3, 0.3},
 PlotPoints → 50, PlotRange → All, AxesLabel → {"x", "t", "u"}]
```

Out[37]=



In[38]:=
```
Plot[u[x, 0.3] /. Flatten[sol], {x, -6, 6}, PlotRange → All]
```

Out[38]=



In[39]:=
```
FindMinimum[u[x, 0.3] /. Flatten[sol], {x, 2.5, 3}]
```

Out[39]=
```
{-4.8575, {x → 3.11458}}
```

In[40]:=
```
FindMinimum[u[x, 0.2] /. Flatten[sol], {x, 1.7, 2.2}]
```

Out[40]=
```
{-4.76587, {x → 2.14086}}
```

The peak moving to the right has a depth of about 5 and a speed of about 10, and is a soliton of the family discussed in the first section. In addition, there are waves moving to the left. These will disperse and lose their form with time.