Report for exercise 5 from group H

Tasks addressed:	5
Authors:	Ahmad Bin Qasim (03693345)
	Kaan Atukalp (03709123)
	Martin Meinel (03710370)
Last compiled:	2020–01–08
Source code:	https://gitlab.lrz.de/ga53rog/praktikum-ml-crowd

The work on tasks was divided in the following way:

Ahmad Bin Qasim (03693345)	Task 1	33%
	Task 2	33%
	Task 3	33%
	Task 4	33%
	Task 5	33%
Kaan Atukalp (03709123)	Task 1	33%
	Task 2	33%
	Task 3	33%
	Task 4	33%
	Task 5	33%
Martin Meinel (03710370)	Task 1	33%
	Task 2	33%
	Task 3	33%
	Task 4	33%
	Task 5	33%

Report on task 1, Approximating functions

Part 1:

For part 1 we loaded the linear data from the "linear_function.txt" file and tried to approximate the function linearly. We used a linear regression model to minimize the mean square error. For this problem there exists a closed form solution to compute the matrix A which contains the parameters to map the input to the output with a minimal mean square error to the original function, we want to approximate.

$$f_{linear}(x) = Ax \in \mathbb{R}^d$$

The minimal mean squares error problem is defined by following equation:

$$\min_{\hat{f}} e(\hat{f}) = \min_{\hat{f}} ||F - \hat{f}(X)||^2 = \min_{A} ||F - XA^T||^2$$

Figure 1 shows the original linear data in blue and its linear approximation in orange. It can be easily seen that all data points are laying on the linear straight which was computed with the closed form solution. The data is linear so the function can be approximated perfectly.

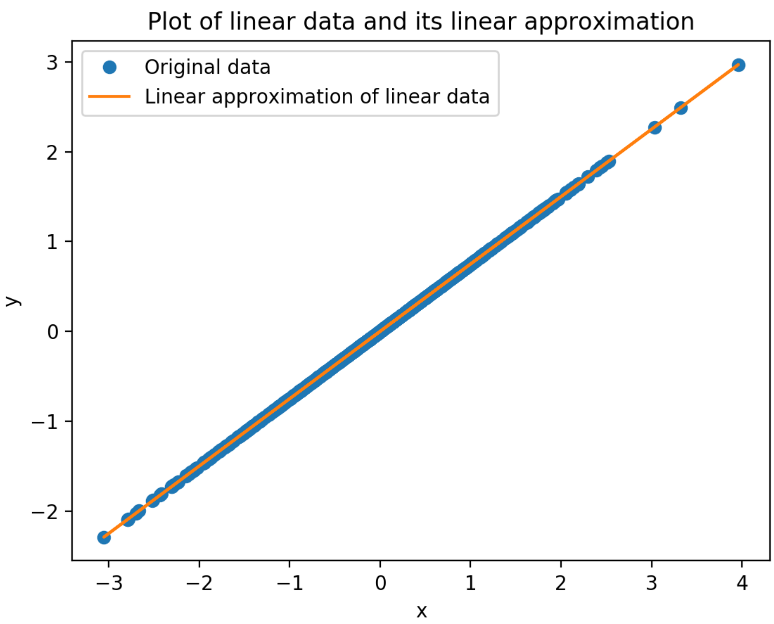

Figure 1: Plot of the linear data and its linear approximation

Part 2:
In the second part we use a nonlinear dataset from the "nonlinear_function.txt" file and tried to approximate the function in a linear way again. Figure 2 shows then original data in blue again. The linear approximation of the function is shown in orange. It can be easily seen that the linear approximation fits very bad. The reason for that is that the function is nonlinear and we try to approximate it linearly.

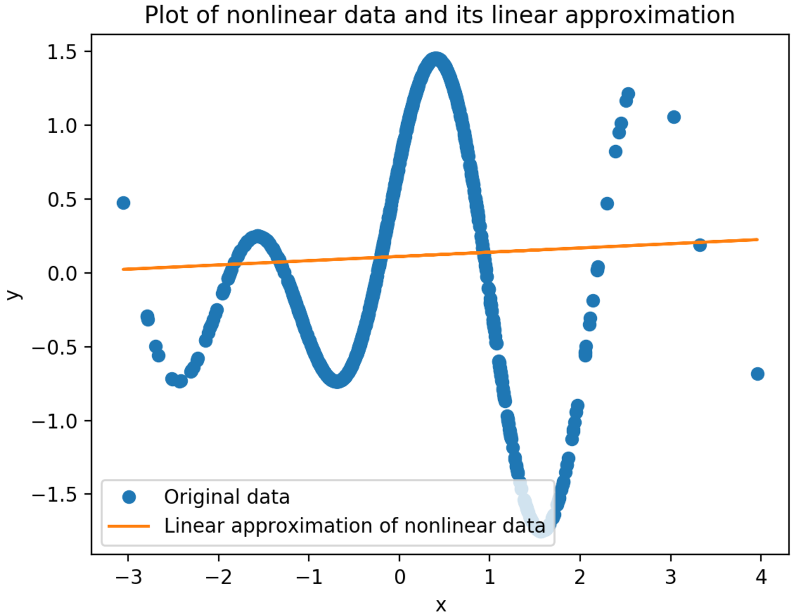

Figure 2: Nonlinear data and its linear approximation

Part 3:
After we tried to approximate the nonlinear dataset with a linear function in part 2, we now try to approximate the unknown nonlinear function f with a combination of radial basis function:

$$f(x) = \sum_{l=1}^{L} c_l \phi_l(x), c_l \in \mathbb{R}^d$$

A radial basis function ϕ is defined as follows:

$$\phi_l(x) = \exp(-||x_l - x||^2/\epsilon^2)$$

where x_l is the center of the basis function and usually just a random point of the data set. There is one x_l for each radial basis function.
The minimal least square error problem looks similar to the linear case:

$$\min_{\hat{f}} e(\hat{f}) = \min_{\hat{f}} ||F - \hat{f}(X)||^2 = \min_{C} ||F - \phi(X)C^T||^2$$

Here the matrix C contains as the matrix A for the linear case the coefficients c_l and

$$\phi(X) := (\phi_1(X), \phi_2(X), \dots, \phi_L(X))$$

We have to choose how many basis functions L we use to approximate the nonlinear function. Besides, we have to choose ϵ appropriately. Figure 3 shows the original data in blue and the approximated function which makes use of radial basis functions in orange. It can be easily seen that the approximation is by far better than the linear approximation of part 2.
In general the approximation fits the original function quite well. We chose ϵ according to the ϵ in the diffusion maps task of the previous exercise sheet. This means we computed the distance matrix among all the x values and took the maximum value for ϵ. This meaximum value is multiplied by 0.05. Besides, we use $L = 15$ radial basis functions for the nonlinear approximation.

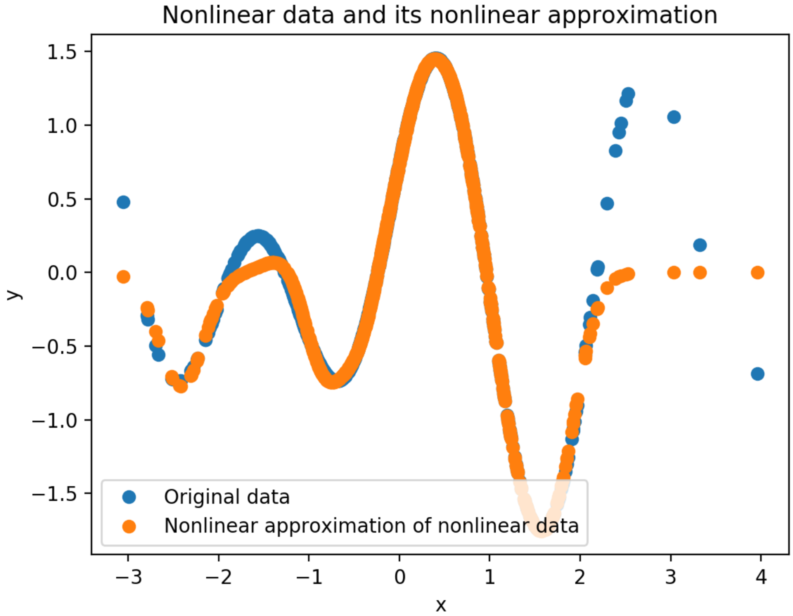

Figure 3: Nonlinear data and its nonlinear approximation with radial basis functions

It is not a good idea to use radial basis functions to approximate a linear function such as the linear data set of A. You can use the radial basis functions to approximate a linear data set, but you should use a linear approximator, because the computational cost of the radial basis function is higher due to the fact that the distance matrix for all x has to be computed to determinate ϵ. Besides the parameters of L and ϵ have to be chosen appropiately to obtain a good approximation.

Report on task 2, First step of a single pedestrian

Part 1:
We use the pandas read_csv function to read the text files containing two dimensional x0 and x1 data. We estimate the vector fields $v^{(k)}$ for each x0, using the finite-difference formula, given hereby:

$$\hat{v}^{(k)} = \frac{x_1^{(k)} - x_0^{(k)}}{\Delta t}$$

Then we approximate the matrix A such that:

$$v(x_0^{(k)}) = v^{(k)} = Ax_0^{(k)}$$

An interesting thing to note is that, bias b is not added in this equation. This will have implications to the prediction later on. It will be mentioned in part 3.

For this purpose, we use the linear approximator implementation from task 1. As mentioned before the goal is to estimate the matrix A, such that the mean square error is minimized. This can be represented through the following equation:

$$\min_{\hat{f}} e(\hat{f}) = \min_{\hat{f}} ||F - \hat{f}(X)||^2 = \min_{A} ||F - XA^T||^2$$

Part 2:
After fitting the linear approximator, we predict the vector fields, $\hat{v}^{(k)}$ for each x0. Then we use $\hat{v}^{(k)}$ with $\Delta t = 0.1$ to calculate $\hat{x1}^{(k)}$ using the formula.

$$\hat{x1}^{(k)} = \hat{v}^{(k)}\Delta t + x0^{(k)}$$

We compare $x1^{(k)}$ and $\hat{x1}^{(k)}$ using the mean squared error formula and obtain the value of: 1.0532185339804091e-13

Part 3:
We set $x0 = (10, 10)^{(0)}$ and then estimate $\hat{v}^{(0)}$. We calculate $\hat{x1}^{(0)}$ using the equation given in part 2. Then using the corresponding $\hat{x1}^{(k-1)}$, we calculate the vector field $\hat{v}^{(k)}$ for it. We repeat this process for $T = 100$. We use steps of $\Delta t = 0.1$, so the total number of iterations are $\frac{T}{\Delta t} = 1000$. As seen in the figure 4, the motion stops at (0,0). We think that reason for this is that, we do not add bias term b to the linear approximator equation. Hence when $x = 0$, then the matrix A has no effect on $\hat{x}$.

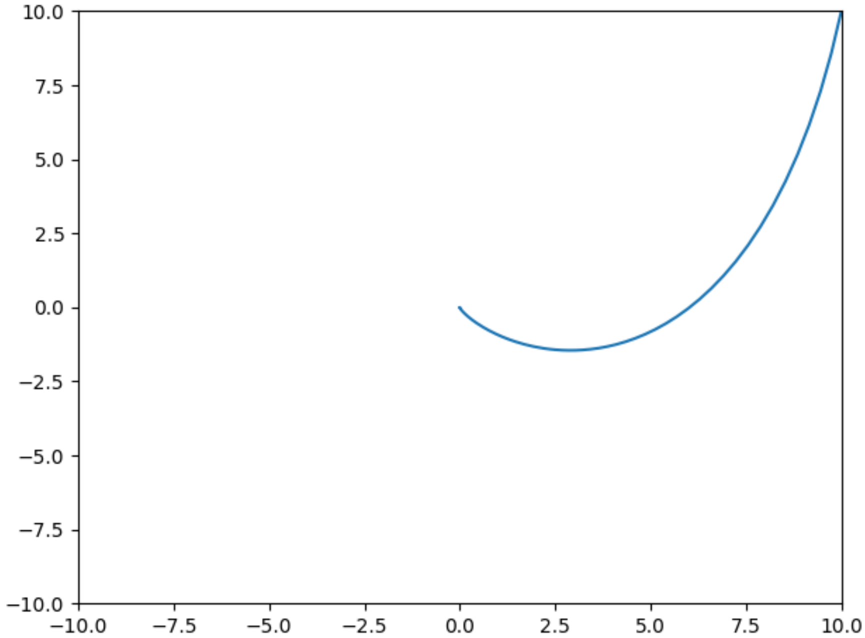

Figure 4: Trajectory of the motion

The phase portrait is given hereby:

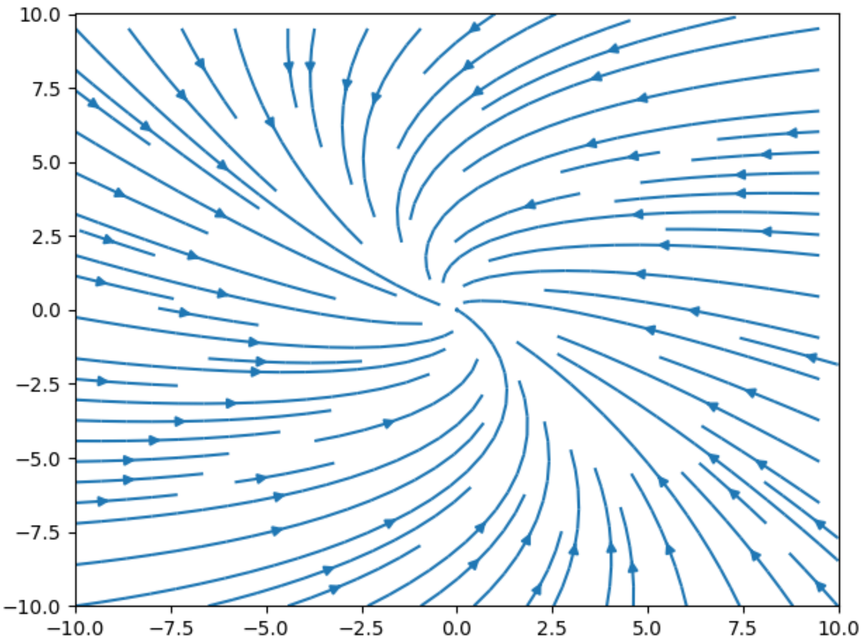

Figure 5: The phase portrait

Report on task 3, Approximating nonlinear vector fields

Part 1:
At first we approximate the vector field $\hat{v}^{(k)}$ for every intial point $x_0^{(k)}$ with the equation:

$$\hat{v}^{(k)} = \frac{x_1^{(k)} - x_0^{(k)}}{\Delta t}$$

Afterwards we try to estimate the vector field describing ψ with a linear operator $A \in \mathbb{R}^{2\times 2}$ such that

$$\frac{\partial}{\partial t}\psi(t,x) \approx \hat{f}_{linear}(x) = Ax$$

After getting A we approximate $\hat{x}_1^{(k)}$ and compute the mean squarred error between the approximated and the known end points for a chosen $\Delta t = 0.5$. The mean squared error is defined as follows:

$$\sum_{k=1}^{N} ||\hat{x}_1^{(k)} - x_1^{(k)}||^2$$

The mean squared error is 75.54.

Part 2:
Part 2 is similar to part 1, but this time we approximate the vector field using radial basis functions such that

$$\frac{\partial}{\partial t}\psi(t,x) \approx \hat{f}_{rbf}(x) = C\phi(X)$$

We choose $L = 10$ and take for ϵ the biggest value of a distance matrix between all initial points x_0. The mean squared error is 0.24.
The mean squarred error is extremely low, so the performance of the nonlinear approximator is very good. Consequently you can see that the vector field is nonlinear. Otherwise the mean squared error of the linear approximation would be lower. You can see that the difference between both mean squared errors is high, because the mean squared error of the nonlinear approximated vector field is very low.

Part 3:
For this part we use the nonlinear approximator, because in our opinion the vector field is nonlinear and therefore the nonlinear approximator shows better performance in comparison to the lineaer approximator. We use the approximated vector field to solve the system for a larger time with all intitial points x_0.
Figure 6 shows the trajectories for the system at the end state. You can see several steady states. The steady states are at the following coordinates:

x-coordinate of steady state	-2.92	3.22	3.65	-4.26
y- coordinate of steady state	3.13	1.85	-1.27	-3.65

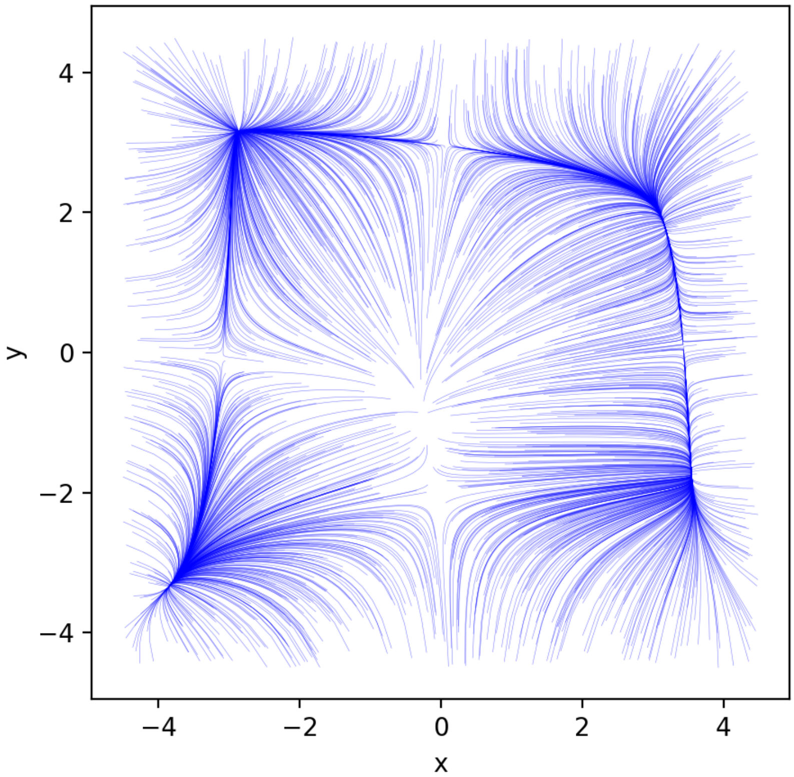

Figure 6: Trajectories of the system at the end state starting at the intial points x_0

The system can be topologically equivalent to a linear system.

Report on task 4, Obstacle avoidance

Pedestrians can avoid obstacles, using Dijkstras algorithm. See figure.

Report on task 5, Tests

TEST1: RiMEA scenario 1 (straight line, ignore premovement time)
- not done, but citing RiMEA guidelines -

TEST2: RiMEA scenario 4 (fundamental diagram, be careful with periodic boundary conditions).
- test successful -

TEST3: RiMEA scenario 6 (movement around a corner).
- test successful -

TEST4: RiMEA scenario
- test successful -