Diff:

Differences between given skeleton and solution

In order to make the sample solution easier to understand, the differences between it and the given skeleton source code were highlighted with the help of the program `diff`.

Legend:

- Gray: unchanged text (only excerpts).

- Green: new lines

- Yellow: changed lines

- Red: deleted lines

Note: Files not listed have not been changed.

This document was created with the help of diff2html erstellt.

```
diff -u ../course04-2d-visualization/exercise/code/01_plot.py ../course04-2d-visualization/exercise/solution/01_plot.py
```

	../course04-2d-visualization/exercise/code/01_plot.py		../course04-2d-visualization/exercise/solution/01_plot.py
21	zz0 = measurment_data[:, 0] # first column of measurement data -> initial value of the simulation	21	zz0 = measurment_data[:, 0] # first column of measurement data -> initial value of the simulation
22		22	
23	# Create empty list for collecting intermediate optimization results:	23	# Create empty list for collecting intermediate optimization results:
24	resList = XXX	24	resList = []
25		25	
26		26	
27	# this is copied from exercise 03.2 (+ one line added)	27	# this is copied from exercise 03.2 (+ one line added)
76	# Exercise 03.1:	76	# Exercise 03.1:
77	# That single line is new:	77	# That single line is new:
78	# save the current simulation result (state array) in the prepared list	78	# save the current simulation result (state array) in the prepared list
79	XXX.append(XXX)	79	resList.append(zz_res)
80		80	
81		81	
82		82	
113		113	
114		114	
115	## After successful saving, insert a 0 here:	115	## After successful saving, insert a 0 here:
116	if 1:	116	if 0:
117	# Optimization - result are the two parameters m2 and l	117	# Optimization - result are the two parameters m2 and l
118		118	
119	# intially guessed values for optimization	119	# intially guessed values for optimization
143	fig = plt.figure(figsize=(300*mm, 170*mm))	143	fig = plt.figure(figsize=(300*mm, 170*mm))
144		144	
145	# subplot for unoptimized curves	145	# subplot for unoptimized curves
146	ax1 = fig.add_subplot(XXX, XXX, XXX)	146	ax1 = fig.add_subplot(3, 1, 1)
147	ax1.plot(tt, XXX[XXX, :], label="measurement")	147	ax1.plot(tt, measurment_data[0, :], label="measurement")
148	ax1.plot(tt, resList[0][XXX, XXX], label="model (initial guess)")	148	ax1.plot(tt, resList[0][0, :], label="model (initial guess)")
149	ax1.legend()	149	ax1.legend()
150	ax1.grid()	150	ax1.grid()
151	ax1.set_ylabel("distance $x ~[m]$")	151	ax1.set_ylabel("distance $x ~[m]$")
152		152	
153		153	
154	# subplot for optimization result	154	# subplot for optimization result
155	ax2 = XXX.add_XXX(XXX, XXX, XXX)	155	ax2 = fig.add_subplot(3, 1, 2)
156	ax2.plot(XXX, XXX, lw=3, label="measurement")	156	ax2.plot(tt, measurment_data[0, :], lw=3, label="measurement")
157	# ...	157	ax2.plot(tt, resList[-1][0, :], label="model (optimized parameters)")
		158	ax2.legend()
		159	ax2.grid()
		160	ax2.set_ylabel("distance $x ~[m]$")
158		161	
159		162	
160	# subplot for remaining error of x position	163	# subplot for remaining error of x position
161	# ...	164	ax3 = fig.add_subplot(3, 1, 3)
162		165	ax3.plot(tt, measurment_data[0, :]-resList[-1][0, :])
		166	ax3.grid()
		167	ax3.set_xlabel("time $t ~[s]$")
		168	ax3.set_ylabel("error ϵ ~[m]")
163		169	
164	plt.savefig("fig1.pdf", bbox_inches="tight")	170	plt.savefig("fig1.pdf", bbox_inches="tight")
165	# plt.show()	171	# plt.show()

```
176    colStep = 0.9 / len(resList)
177    gray_value = str(0.01 + (0.9 - i*colStep))
178
179    plt.plot(XXX, XXX[XXX][XXX, XXX], color=gray_value )
180
181# Plotting the measurement and the simulation
182plt.plot(XXX, XXX, color="#3366FF", lw=4, label="measurement")
183plt.plot(XXX, XXX[-1][XXX, XXX], color="#FF9900", ls="--", lw=2, label="model (optimized
   parameters)")
184
185# grid
186# legend
187# x label
188# y label
189
190
191# here we use png because due to the many individual lines the pdf would be quite big and slow
192plt.savefig("fig2.png", bbox_inches="tight")
⋮
211
212# overview on color maps:
213# https://matplotlib.org/examples/color/colormaps_reference.html
214ax.plot_surface(XXX, rstride=1, cstride=1, cmap=cm.XXX)
215ax.set_xlabel("x")
216ax.set_ylabel("y")
217ax.set_zlabel("z")
```

```
182    colStep = 0.9 / len(resList)
183    gray_value = str(0.01 + (0.9 - i*colStep))
184
185    plt.plot(tt, resList[i][0, :], color=gray_value )
186
187# Plotting the measurement and the simulation
188plt.plot(tt, measurment_data[0, :], color="#3366FF", lw=4, label="measurement")
189plt.plot(tt, resList[-1][0, :], color="#FF9900", ls="--", lw=2, label="model (optimized
   parameters)")
190plt.grid()
191plt.legend()
192plt.xlabel("time $t ~[s]$")
193plt.ylabel("distance $x ~[m]$")
194
195# here we use png because due to the many individual lines the pdf would be quite big and
   slow
196plt.savefig("fig2.png", bbox_inches="tight")
⋮
215
216# overview on color maps:
217# https://matplotlib.org/examples/color/colormaps_reference.html
218ax.plot_surface(X, Y, Z, rstride=1, cstride=1, cmap=cm.viridis)
219ax.set_xlabel("x")
220ax.set_ylabel("y")
221ax.set_zlabel("z")
```

Nur in ../course04-2d-visualization/exercise/solution/: fig1.pdf.
Nur in ../course04-2d-visualization/exercise/solution/: fig2.png.
Nur in ../course04-2d-visualization/exercise/solution/: fig3.png.
Nur in ../course04-2d-visualization/exercise/solution/: __pycache__.
Nur in ../course04-2d-visualization/exercise/solution/: res_list.pcl.