**Project description**

1. State the region and the domain category that your data sets are about (e.g., Chaohu, China and sports or athletics).

United States and gender inequality of labor force participation rate and its trend

1. Create a research question about the domain category and region that you identified.

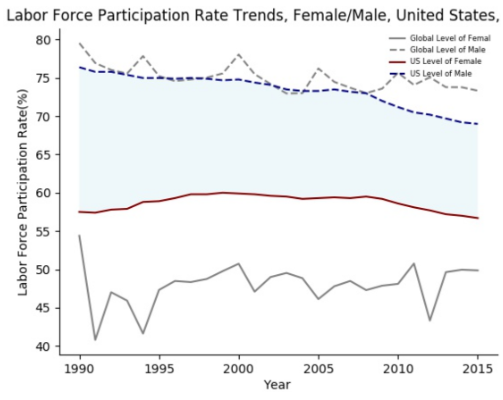
What's the gender inequality of labor force participation rate and its trends from 1990 - 2015 in United States, compared with global level.

1. Provide at least two links to publicly accessible datasets. These could be links to files such as CSV or Excel files, or links to websites which might have data in tabular form, such as Wikipedia pages.

Note： you can access to links below only when you signup on the worldbank website  
  
Labor force participation rate(ages 15 and older as long as they have ability to work), female  
<http://databank.worldbank.org/data/Labor-force-participation-rate%EF%BC%8Cfemale/id/6f6afc4c>  
  
Labor force participation rate(ages 15 and older as long as they have ability to work), male[http://databank.worldbank.org/data/Labor-force-participation-rate，male/id/560f5316](http://databank.worldbank.org/data/Labor-force-participation-rate%EF%BC%8Cmale/id/560f5316)  
  
Population ages 15-64, female[http://databank.worldbank.org/data/Population-ages-15-64，femal/id/5c668b8a](http://databank.worldbank.org/data/Population-ages-15-64%EF%BC%8Cfemal/id/5c668b8a)  
Population males 15-64, male[http://databank.worldbank.org/data/Population-ages-15-64，male/id/b163f56d](http://databank.worldbank.org/data/Population-ages-15-64%EF%BC%8Cmale/id/b163f56d)

1. Upload an image which addresses your research question.

In addition to addressing your research question, this visual should address Cairo's principles of truthfulness, functionality, beauty, and insightfulness.



1. Provide a short (1-2 paragraphs) justification of how your visual addresses your research question.

This visual use line plot shows the labor force participation rate trends of female and male from 1990-2015 in United States and global average level. The gender inequality of female and male can be seen by comparing the dash line and solid line. To be more specific, two dash lines represents the male labor force participation rate in US and global. Two solid lines represents the female labor force participation rate in US and global. From these two kinds of lines, we can conclude that the inequality of labor force participation rate of different gender did exist in United States and the whole world.

Also, the line plot shows the trend of labor force participation rate from 1990 to 2015 in US (the darkblue line for male, and darkred line for femal). Also, the gray lines provide a global level (calculated by weighted average of all countries) of labor force participation rate of female and male. We can see that, for United States, the gap (shadow area) between female LFPR and male LFPR were lower and lower from 1990 to 2015. Moreover, the LFPR of female in United States were always higher than global average.

1. As this assignment is for the whole course, you must incorporate and defend the principles discussed in the first week, specifically, Cairo’s principles of truth, beauty, function, and insight.

For each of the following prompts, please provide a response that links each principle to one or more elements of your visual.

* Describe your design choices for your visual in regards to Cairo's principle of **truthfulness**.
* Describe your design choices for your visual in regards to Cairo's principle of **beauty**.
* Describe your design choices for your visual in regards to Cairo's principle of **functionality**.
* Describe your design choices for your visual in regards to Cairo's principle of **insightfulness**.

**Truthfulness:** I collect data from reliable data source, the WorldBank. Also, in order to provide a global level of labor force participation rate, I also collect the female and male population data (ages from 15 to 64) of each country . I use these data set to calculate the weighted average of labor force participation rate, which is better than directly average the LFPR of all countries. Also, I provide clear axis, label, legend in this visual, which avoid misleading.

**Beauty:** I use different lines (dash or solid) to represent gender, and highlight the US level by plot the global level as gray line. Also, I shade the gap of female level and male level in US to high light the inequality and the change of inequality. In order to show the trend more clearly, I adjusted the minimum value of y axis. In this way, the change of the trends will be more clear.

**Functionality:** To clear the junk of this plot, I removed the top and right line of frame. Also, I adjusted the minimum value of y-axis. I chose the line plot which is suitable to show the trends. Besides, the shadow area help reader to see the inequality and its trend more clearly.

**Insightfulness:** This visual provide a lot of useful information of female/male labor force participation rate trend from 1990 to 2015 in US and the whole world. First, we can get that the inequality labor force participation rate of gender did exist in US and the whole world. Second, in US, female are more willing to world compared with the whole world， which is consistent to the strong feminism in US. Third, the gap between the female and male labor force participation rate was smaller and smaller in US, showing that the inequality of LFPR in gender was smaller and smaller.

1. Provide Source Code:
2. {
3. "cells": [
4. {
5. "cell\_type": "markdown",
6. "metadata": {},
7. "source": [
8. "# Assignment 4\n",
9. "\n",
10. "Before working on this assignment please read these instructions fully. In the submission area, you will notice that you can click the link to \*\*Preview the Grading\*\* for each step of the assignment. This is the criteria that will be used for peer grading. Please familiarize yourself with the criteria before beginning the assignment.\n",
11. "\n",
12. "This assignment requires that you to find \*\*at least\*\* two datasets on the web which are related, and that you visualize these datasets to answer a question with the broad topic of \*\*weather phenomena\*\* (see below) for the region of \*\*Ann Arbor, Michigan, United States\*\*, or \*\*United States\*\* more broadly.\n",
13. "\n",
14. "You can merge these datasets with data from different regions if you like! For instance, you might want to compare \*\*Ann Arbor, Michigan, United States\*\* to Ann Arbor, USA. In that case at least one source file must be about \*\*Ann Arbor, Michigan, United States\*\*.\n",
15. "\n",
16. "You are welcome to choose datasets at your discretion, but keep in mind \*\*they will be shared with your peers\*\*, so choose appropriate datasets. Sensitive, confidential, illicit, and proprietary materials are not good choices for datasets for this assignment. You are welcome to upload datasets of your own as well, and link to them using a third party repository such as github, bitbucket, pastebin, etc. Please be aware of the Coursera terms of service with respect to intellectual property.\n",
17. "\n",
18. "Also, you are welcome to preserve data in its original language, but for the purposes of grading you should provide english translations. You are welcome to provide multiple visuals in different languages if you would like!\n",
19. "\n",
20. "As this assignment is for the whole course, you must incorporate principles discussed in the first week, such as having as high data-ink ratio (Tufte) and aligning with Cairoâ€™s principles of truth, beauty, function, and insight.\n",
21. "\n",
22. "Here are the assignment instructions:\n",
23. "\n",
24. " \* State the region and the domain category that your data sets are about (e.g., \*\*Ann Arbor, Michigan, United States\*\* and \*\*weather phenomena\*\*).\n",
25. " \* You must state a question about the domain category and region that you identified as being interesting.\n",
26. " \* You must provide at least two links to available datasets. These could be links to files such as CSV or Excel files, or links to websites which might have data in tabular form, such as Wikipedia pages.\n",
27. " \* You must upload an image which addresses the research question you stated. In addition to addressing the question, this visual should follow Cairo's principles of truthfulness, functionality, beauty, and insightfulness.\n",
28. " \* You must contribute a short (1-2 paragraph) written justification of how your visualization addresses your stated research question.\n",
29. "\n",
30. "What do we mean by \*\*weather phenomena\*\*? For this category you might want to consider seasonal changes, natural disasters, or historical trends.\n",
31. "\n",
32. "## Tips\n",
33. "\* Wikipedia is an excellent source of data, and I strongly encourage you to explore it for new data sources.\n",
34. "\* Many governments run open data initiatives at the city, region, and country levels, and these are wonderful resources for localized data sources.\n",
35. "\* Several international agencies, such as the [United Nations](http://data.un.org/), the [World Bank](http://data.worldbank.org/), the [Global Open Data Index](http://index.okfn.org/place/) are other great places to look for data.\n",
36. "\* This assignment requires you to convert and clean datafiles. Check out the discussion forums for tips on how to do this from various sources, and share your successes with your fellow students!\n",
37. "\n",
38. "## Example\n",
39. "Looking for an example? Here's what our course assistant put together for the \*\*Ann Arbor, MI, USA\*\* area using \*\*sports and athletics\*\* as the topic. [Example Solution File](./readonly/Assignment4\_example.pdf)"
40. ]
41. },
42. {
43. "cell\_type": "code",
44. "execution\_count": 1,
45. "metadata": {
46. "collapsed": true
47. },
48. "outputs": [],
49. "source": [
50. "% matplotlib notebook\n",
51. "import pandas as pd\n",
52. "import numpy as np\n",
53. "import matplotlib.pyplot as plt"
54. ]
55. },
56. {
57. "cell\_type": "code",
58. "execution\_count": 2,
59. "metadata": {
60. "collapsed": false,
61. "scrolled": true
62. },
63. "outputs": [],
64. "source": [
65. "# import data\n",
66. "LFPR\_FM = pd.read\_excel('LFPR\_FM.xls', skip\_footer = 5).drop(labels = ['Series Name', 2016], axis = 1).set\_index('Country Name')\n",
67. "LFPR\_MA = pd.read\_excel('LFPR\_MA.xls', skip\_footer = 5).drop(labels = ['Series Name', 2016], axis = 1).set\_index('Country Name')\n",
68. "Pop\_FM = pd.read\_excel('Pop\_FM.xls', skip\_footer = 5).drop(labels = ['Series Name', 2016], axis = 1).set\_index('Country Name').dropna()\n",
69. "Pop\_MA = pd.read\_excel('Pop\_MA.xls', skip\_footer = 5).drop(labels = ['Series Name', 2016], axis = 1).set\_index('Country Name').dropna()"
70. ]
71. },
72. {
73. "cell\_type": "code",
74. "execution\_count": null,
75. "metadata": {
76. "collapsed": false
77. },
78. "outputs": [],
79. "source": [
80. "#LFPR\_FM.head()"
81. ]
82. },
83. {
84. "cell\_type": "code",
85. "execution\_count": 3,
86. "metadata": {
87. "collapsed": false
88. },
89. "outputs": [],
90. "source": [
91. "## calculate weight of Pop\n",
92. "SumF = Pop\_FM.sum(axis = 0)\n",
93. "\n",
94. "for i in range(len(Pop\_FM.T)):\n",
95. " Pop\_FM.T.iloc[i] = Pop\_FM.T.iloc[i]/SumF.values[i]\n",
96. "\n",
97. "SumM = Pop\_MA.sum(axis = 0)\n",
98. "\n",
99. "for i in range(len(Pop\_MA.T)):\n",
100. " Pop\_MA.T.iloc[i] = Pop\_MA.T.iloc[i]/SumM.values[i]"
101. ]
102. },
103. {
104. "cell\_type": "code",
105. "execution\_count": null,
106. "metadata": {
107. "collapsed": true
108. },
109. "outputs": [],
110. "source": [
111. "## Calculate weighted average of LFPR for female"
112. ]
113. },
114. {
115. "cell\_type": "code",
116. "execution\_count": 7,
117. "metadata": {
118. "collapsed": false,
119. "scrolled": false
120. },
121. "outputs": [],
122. "source": [
123. "LFPR = pd.DataFrame(columns = LFPR\_FM.columns)\n",
124. "LFPR.loc['Global\_FM'] = (LFPR\_FM\*Pop\_FM).sum(axis = 0) "
125. ]
126. },
127. {
128. "cell\_type": "code",
129. "execution\_count": 8,
130. "metadata": {
131. "collapsed": false,
132. "scrolled": true
133. },
134. "outputs": [],
135. "source": [
136. "for col in range (1990, 2016):\n",
137. " notna = (LFPR\_FM[col].dropna()).index\n",
138. " Sum\_W = (Pop\_FM[col].loc[notna]).sum()\n",
139. " LFPR[col].loc['Global\_FM'] = (LFPR[col].loc['Global\_FM'])/Sum\_W"
140. ]
141. },
142. {
143. "cell\_type": "code",
144. "execution\_count": null,
145. "metadata": {
146. "collapsed": true
147. },
148. "outputs": [],
149. "source": [
150. "## Calculate weighted average of LFPR for male"
151. ]
152. },
153. {
154. "cell\_type": "code",
155. "execution\_count": 9,
156. "metadata": {
157. "collapsed": false,
158. "scrolled": true
159. },
160. "outputs": [],
161. "source": [
162. "LFPR.loc['Global\_MA'] = (LFPR\_MA\*Pop\_MA).sum(axis = 0) "
163. ]
164. },
165. {
166. "cell\_type": "code",
167. "execution\_count": 10,
168. "metadata": {
169. "collapsed": true
170. },
171. "outputs": [],
172. "source": [
173. "for col in range (1990, 2016):\n",
174. " notna = (LFPR\_MA[col].dropna()).index\n",
175. " Sum\_W = (Pop\_MA[col].loc[notna]).sum()\n",
176. " LFPR[col].loc['Global\_MA'] = (LFPR[col].loc['Global\_MA'])/Sum\_W"
177. ]
178. },
179. {
180. "cell\_type": "code",
181. "execution\_count": null,
182. "metadata": {
183. "collapsed": true
184. },
185. "outputs": [],
186. "source": [
187. "## add us data in dataframe LFPR"
188. ]
189. },
190. {
191. "cell\_type": "code",
192. "execution\_count": 11,
193. "metadata": {
194. "collapsed": false
195. },
196. "outputs": [],
197. "source": [
198. "LFPR.loc['US\_FM'] = LFPR\_FM.loc['United States']\n",
199. "LFPR.loc['US\_MA'] = LFPR\_MA.loc['United States']"
200. ]
201. },
202. {
203. "cell\_type": "code",
204. "execution\_count": 13,
205. "metadata": {
206. "collapsed": false
207. },
208. "outputs": [
209. {
210. "data": {
211. "text/html": [
212. "<div>\n",
213. "<table border=\"1\" class=\"dataframe\">\n",
214. " <thead>\n",
215. " <tr style=\"text-align: right;\">\n",
216. " <th></th>\n",
217. " <th>1990</th>\n",
218. " <th>1991</th>\n",
219. " <th>1992</th>\n",
220. " <th>1993</th>\n",
221. " <th>1994</th>\n",
222. " <th>1995</th>\n",
223. " <th>1996</th>\n",
224. " <th>1997</th>\n",
225. " <th>1998</th>\n",
226. " <th>1999</th>\n",
227. " <th>...</th>\n",
228. " <th>2006</th>\n",
229. " <th>2007</th>\n",
230. " <th>2008</th>\n",
231. " <th>2009</th>\n",
232. " <th>2010</th>\n",
233. " <th>2011</th>\n",
234. " <th>2012</th>\n",
235. " <th>2013</th>\n",
236. " <th>2014</th>\n",
237. " <th>2015</th>\n",
238. " </tr>\n",
239. " </thead>\n",
240. " <tbody>\n",
241. " <tr>\n",
242. " <th>Global\_FM</th>\n",
243. " <td>54.386401</td>\n",
244. " <td>40.781028</td>\n",
245. " <td>47.006974</td>\n",
246. " <td>45.915798</td>\n",
247. " <td>41.611729</td>\n",
248. " <td>47.309024</td>\n",
249. " <td>48.477503</td>\n",
250. " <td>48.335934</td>\n",
251. " <td>48.744350</td>\n",
252. " <td>49.764385</td>\n",
253. " <td>...</td>\n",
254. " <td>47.786652</td>\n",
255. " <td>48.480763</td>\n",
256. " <td>47.287586</td>\n",
257. " <td>47.863166</td>\n",
258. " <td>48.102283</td>\n",
259. " <td>50.760268</td>\n",
260. " <td>43.307749</td>\n",
261. " <td>49.631321</td>\n",
262. " <td>49.947030</td>\n",
263. " <td>49.858233</td>\n",
264. " </tr>\n",
265. " <tr>\n",
266. " <th>Global\_MA</th>\n",
267. " <td>79.549186</td>\n",
268. " <td>76.950327</td>\n",
269. " <td>76.046112</td>\n",
270. " <td>75.580993</td>\n",
271. " <td>77.856944</td>\n",
272. " <td>75.242196</td>\n",
273. " <td>74.597839</td>\n",
274. " <td>74.807830</td>\n",
275. " <td>75.046100</td>\n",
276. " <td>75.589439</td>\n",
277. " <td>...</td>\n",
278. " <td>74.486670</td>\n",
279. " <td>73.718535</td>\n",
280. " <td>73.021996</td>\n",
281. " <td>73.603670</td>\n",
282. " <td>75.670644</td>\n",
283. " <td>74.089032</td>\n",
284. " <td>75.083727</td>\n",
285. " <td>73.795009</td>\n",
286. " <td>73.785087</td>\n",
287. " <td>73.336234</td>\n",
288. " </tr>\n",
289. " <tr>\n",
290. " <th>US\_FM</th>\n",
291. " <td>57.500000</td>\n",
292. " <td>57.400002</td>\n",
293. " <td>57.799999</td>\n",
294. " <td>57.900002</td>\n",
295. " <td>58.799999</td>\n",
296. " <td>58.900002</td>\n",
297. " <td>59.299999</td>\n",
298. " <td>59.799999</td>\n",
299. " <td>59.799999</td>\n",
300. " <td>60.000000</td>\n",
301. " <td>...</td>\n",
302. " <td>59.400002</td>\n",
303. " <td>59.299999</td>\n",
304. " <td>59.500000</td>\n",
305. " <td>59.200001</td>\n",
306. " <td>58.599998</td>\n",
307. " <td>58.099998</td>\n",
308. " <td>57.700001</td>\n",
309. " <td>57.200001</td>\n",
310. " <td>57.000000</td>\n",
311. " <td>56.700001</td>\n",
312. " </tr>\n",
313. " <tr>\n",
314. " <th>US\_MA</th>\n",
315. " <td>76.400002</td>\n",
316. " <td>75.800003</td>\n",
317. " <td>75.800003</td>\n",
318. " <td>75.400002</td>\n",
319. " <td>75.000000</td>\n",
320. " <td>75.000000</td>\n",
321. " <td>74.900002</td>\n",
322. " <td>75.000000</td>\n",
323. " <td>74.900002</td>\n",
324. " <td>74.699997</td>\n",
325. " <td>...</td>\n",
326. " <td>73.500000</td>\n",
327. " <td>73.199997</td>\n",
328. " <td>73.000000</td>\n",
329. " <td>72.000000</td>\n",
330. " <td>71.199997</td>\n",
331. " <td>70.500000</td>\n",
332. " <td>70.199997</td>\n",
333. " <td>69.699997</td>\n",
334. " <td>69.199997</td>\n",
335. " <td>69.000000</td>\n",
336. " </tr>\n",
337. " </tbody>\n",
338. "</table>\n",
339. "<p>4 rows Ã— 26 columns</p>\n",
340. "</div>"
341. ],
342. "text/plain": [
343. " 1990 1991 1992 1993 1994 1995 \\\n",
344. "Global\_FM 54.386401 40.781028 47.006974 45.915798 41.611729 47.309024 \n",
345. "Global\_MA 79.549186 76.950327 76.046112 75.580993 77.856944 75.242196 \n",
346. "US\_FM 57.500000 57.400002 57.799999 57.900002 58.799999 58.900002 \n",
347. "US\_MA 76.400002 75.800003 75.800003 75.400002 75.000000 75.000000 \n",
348. "\n",
349. " 1996 1997 1998 1999 ... 2006 \\\n",
350. "Global\_FM 48.477503 48.335934 48.744350 49.764385 ... 47.786652 \n",
351. "Global\_MA 74.597839 74.807830 75.046100 75.589439 ... 74.486670 \n",
352. "US\_FM 59.299999 59.799999 59.799999 60.000000 ... 59.400002 \n",
353. "US\_MA 74.900002 75.000000 74.900002 74.699997 ... 73.500000 \n",
354. "\n",
355. " 2007 2008 2009 2010 2011 2012 \\\n",
356. "Global\_FM 48.480763 47.287586 47.863166 48.102283 50.760268 43.307749 \n",
357. "Global\_MA 73.718535 73.021996 73.603670 75.670644 74.089032 75.083727 \n",
358. "US\_FM 59.299999 59.500000 59.200001 58.599998 58.099998 57.700001 \n",
359. "US\_MA 73.199997 73.000000 72.000000 71.199997 70.500000 70.199997 \n",
360. "\n",
361. " 2013 2014 2015 \n",
362. "Global\_FM 49.631321 49.947030 49.858233 \n",
363. "Global\_MA 73.795009 73.785087 73.336234 \n",
364. "US\_FM 57.200001 57.000000 56.700001 \n",
365. "US\_MA 69.699997 69.199997 69.000000 \n",
366. "\n",
367. "[4 rows x 26 columns]"
368. ]
369. },
370. "execution\_count": 13,
371. "metadata": {},
372. "output\_type": "execute\_result"
373. }
374. ],
375. "source": [
376. "LFPR"
377. ]
378. },
379. {
380. "cell\_type": "code",
381. "execution\_count": 26,
382. "metadata": {
383. "collapsed": false
384. },
385. "outputs": [
386. {
387. "data": {
388. "application/javascript": [
389. "/\* Put everything inside the global mpl namespace \*/\n",
390. "window.mpl = {};\n",
391. "\n",
392. "\n",
393. "mpl.get\_websocket\_type = function() {\n",
394. " if (typeof(WebSocket) !== 'undefined') {\n",
395. " return WebSocket;\n",
396. " } else if (typeof(MozWebSocket) !== 'undefined') {\n",
397. " return MozWebSocket;\n",
398. " } else {\n",
399. " alert('Your browser does not have WebSocket support.' +\n",
400. " 'Please try Chrome, Safari or Firefox â‰¥ 6. ' +\n",
401. " 'Firefox 4 and 5 are also supported but you ' +\n",
402. " 'have to enable WebSockets in about:config.');\n",
403. " };\n",
404. "}\n",
405. "\n",
406. "mpl.figure = function(figure\_id, websocket, ondownload, parent\_element) {\n",
407. " this.id = figure\_id;\n",
408. "\n",
409. " this.ws = websocket;\n",
410. "\n",
411. " this.supports\_binary = (this.ws.binaryType != undefined);\n",
412. "\n",
413. " if (!this.supports\_binary) {\n",
414. " var warnings = document.getElementById(\"mpl-warnings\");\n",
415. " if (warnings) {\n",
416. " warnings.style.display = 'block';\n",
417. " warnings.textContent = (\n",
418. " \"This browser does not support binary websocket messages. \" +\n",
419. " \"Performance may be slow.\");\n",
420. " }\n",
421. " }\n",
422. "\n",
423. " this.imageObj = new Image();\n",
424. "\n",
425. " this.context = undefined;\n",
426. " this.message = undefined;\n",
427. " this.canvas = undefined;\n",
428. " this.rubberband\_canvas = undefined;\n",
429. " this.rubberband\_context = undefined;\n",
430. " this.format\_dropdown = undefined;\n",
431. "\n",
432. " this.image\_mode = 'full';\n",
433. "\n",
434. " this.root = $('<div/>');\n",
435. " this.\_root\_extra\_style(this.root)\n",
436. " this.root.attr('style', 'display: inline-block');\n",
437. "\n",
438. " $(parent\_element).append(this.root);\n",
439. "\n",
440. " this.\_init\_header(this);\n",
441. " this.\_init\_canvas(this);\n",
442. " this.\_init\_toolbar(this);\n",
443. "\n",
444. " var fig = this;\n",
445. "\n",
446. " this.waiting = false;\n",
447. "\n",
448. " this.ws.onopen = function () {\n",
449. " fig.send\_message(\"supports\_binary\", {value: fig.supports\_binary});\n",
450. " fig.send\_message(\"send\_image\_mode\", {});\n",
451. " if (mpl.ratio != 1) {\n",
452. " fig.send\_message(\"set\_dpi\_ratio\", {'dpi\_ratio': mpl.ratio});\n",
453. " }\n",
454. " fig.send\_message(\"refresh\", {});\n",
455. " }\n",
456. "\n",
457. " this.imageObj.onload = function() {\n",
458. " if (fig.image\_mode == 'full') {\n",
459. " // Full images could contain transparency (where diff images\n",
460. " // almost always do), so we need to clear the canvas so that\n",
461. " // there is no ghosting.\n",
462. " fig.context.clearRect(0, 0, fig.canvas.width, fig.canvas.height);\n",
463. " }\n",
464. " fig.context.drawImage(fig.imageObj, 0, 0);\n",
465. " };\n",
466. "\n",
467. " this.imageObj.onunload = function() {\n",
468. " this.ws.close();\n",
469. " }\n",
470. "\n",
471. " this.ws.onmessage = this.\_make\_on\_message\_function(this);\n",
472. "\n",
473. " this.ondownload = ondownload;\n",
474. "}\n",
475. "\n",
476. "mpl.figure.prototype.\_init\_header = function() {\n",
477. " var titlebar = $(\n",
478. " '<div class=\"ui-dialog-titlebar ui-widget-header ui-corner-all ' +\n",
479. " 'ui-helper-clearfix\"/>');\n",
480. " var titletext = $(\n",
481. " '<div class=\"ui-dialog-title\" style=\"width: 100%; ' +\n",
482. " 'text-align: center; padding: 3px;\"/>');\n",
483. " titlebar.append(titletext)\n",
484. " this.root.append(titlebar);\n",
485. " this.header = titletext[0];\n",
486. "}\n",
487. "\n",
488. "\n",
489. "\n",
490. "mpl.figure.prototype.\_canvas\_extra\_style = function(canvas\_div) {\n",
491. "\n",
492. "}\n",
493. "\n",
494. "\n",
495. "mpl.figure.prototype.\_root\_extra\_style = function(canvas\_div) {\n",
496. "\n",
497. "}\n",
498. "\n",
499. "mpl.figure.prototype.\_init\_canvas = function() {\n",
500. " var fig = this;\n",
501. "\n",
502. " var canvas\_div = $('<div/>');\n",
503. "\n",
504. " canvas\_div.attr('style', 'position: relative; clear: both; outline: 0');\n",
505. "\n",
506. " function canvas\_keyboard\_event(event) {\n",
507. " return fig.key\_event(event, event['data']);\n",
508. " }\n",
509. "\n",
510. " canvas\_div.keydown('key\_press', canvas\_keyboard\_event);\n",
511. " canvas\_div.keyup('key\_release', canvas\_keyboard\_event);\n",
512. " this.canvas\_div = canvas\_div\n",
513. " this.\_canvas\_extra\_style(canvas\_div)\n",
514. " this.root.append(canvas\_div);\n",
515. "\n",
516. " var canvas = $('<canvas/>');\n",
517. " canvas.addClass('mpl-canvas');\n",
518. " canvas.attr('style', \"left: 0; top: 0; z-index: 0; outline: 0\")\n",
519. "\n",
520. " this.canvas = canvas[0];\n",
521. " this.context = canvas[0].getContext(\"2d\");\n",
522. "\n",
523. " var backingStore = this.context.backingStorePixelRatio ||\n",
524. "\tthis.context.webkitBackingStorePixelRatio ||\n",
525. "\tthis.context.mozBackingStorePixelRatio ||\n",
526. "\tthis.context.msBackingStorePixelRatio ||\n",
527. "\tthis.context.oBackingStorePixelRatio ||\n",
528. "\tthis.context.backingStorePixelRatio || 1;\n",
529. "\n",
530. " mpl.ratio = (window.devicePixelRatio || 1) / backingStore;\n",
531. "\n",
532. " var rubberband = $('<canvas/>');\n",
533. " rubberband.attr('style', \"position: absolute; left: 0; top: 0; z-index: 1;\")\n",
534. "\n",
535. " var pass\_mouse\_events = true;\n",
536. "\n",
537. " canvas\_div.resizable({\n",
538. " start: function(event, ui) {\n",
539. " pass\_mouse\_events = false;\n",
540. " },\n",
541. " resize: function(event, ui) {\n",
542. " fig.request\_resize(ui.size.width, ui.size.height);\n",
543. " },\n",
544. " stop: function(event, ui) {\n",
545. " pass\_mouse\_events = true;\n",
546. " fig.request\_resize(ui.size.width, ui.size.height);\n",
547. " },\n",
548. " });\n",
549. "\n",
550. " function mouse\_event\_fn(event) {\n",
551. " if (pass\_mouse\_events)\n",
552. " return fig.mouse\_event(event, event['data']);\n",
553. " }\n",
554. "\n",
555. " rubberband.mousedown('button\_press', mouse\_event\_fn);\n",
556. " rubberband.mouseup('button\_release', mouse\_event\_fn);\n",
557. " // Throttle sequential mouse events to 1 every 20ms.\n",
558. " rubberband.mousemove('motion\_notify', mouse\_event\_fn);\n",
559. "\n",
560. " rubberband.mouseenter('figure\_enter', mouse\_event\_fn);\n",
561. " rubberband.mouseleave('figure\_leave', mouse\_event\_fn);\n",
562. "\n",
563. " canvas\_div.on(\"wheel\", function (event) {\n",
564. " event = event.originalEvent;\n",
565. " event['data'] = 'scroll'\n",
566. " if (event.deltaY < 0) {\n",
567. " event.step = 1;\n",
568. " } else {\n",
569. " event.step = -1;\n",
570. " }\n",
571. " mouse\_event\_fn(event);\n",
572. " });\n",
573. "\n",
574. " canvas\_div.append(canvas);\n",
575. " canvas\_div.append(rubberband);\n",
576. "\n",
577. " this.rubberband = rubberband;\n",
578. " this.rubberband\_canvas = rubberband[0];\n",
579. " this.rubberband\_context = rubberband[0].getContext(\"2d\");\n",
580. " this.rubberband\_context.strokeStyle = \"#000000\";\n",
581. "\n",
582. " this.\_resize\_canvas = function(width, height) {\n",
583. " // Keep the size of the canvas, canvas container, and rubber band\n",
584. " // canvas in synch.\n",
585. " canvas\_div.css('width', width)\n",
586. " canvas\_div.css('height', height)\n",
587. "\n",
588. " canvas.attr('width', width \* mpl.ratio);\n",
589. " canvas.attr('height', height \* mpl.ratio);\n",
590. " canvas.attr('style', 'width: ' + width + 'px; height: ' + height + 'px;');\n",
591. "\n",
592. " rubberband.attr('width', width);\n",
593. " rubberband.attr('height', height);\n",
594. " }\n",
595. "\n",
596. " // Set the figure to an initial 600x600px, this will subsequently be updated\n",
597. " // upon first draw.\n",
598. " this.\_resize\_canvas(600, 600);\n",
599. "\n",
600. " // Disable right mouse context menu.\n",
601. " $(this.rubberband\_canvas).bind(\"contextmenu\",function(e){\n",
602. " return false;\n",
603. " });\n",
604. "\n",
605. " function set\_focus () {\n",
606. " canvas.focus();\n",
607. " canvas\_div.focus();\n",
608. " }\n",
609. "\n",
610. " window.setTimeout(set\_focus, 100);\n",
611. "}\n",
612. "\n",
613. "mpl.figure.prototype.\_init\_toolbar = function() {\n",
614. " var fig = this;\n",
615. "\n",
616. " var nav\_element = $('<div/>')\n",
617. " nav\_element.attr('style', 'width: 100%');\n",
618. " this.root.append(nav\_element);\n",
619. "\n",
620. " // Define a callback function for later on.\n",
621. " function toolbar\_event(event) {\n",
622. " return fig.toolbar\_button\_onclick(event['data']);\n",
623. " }\n",
624. " function toolbar\_mouse\_event(event) {\n",
625. " return fig.toolbar\_button\_onmouseover(event['data']);\n",
626. " }\n",
627. "\n",
628. " for(var toolbar\_ind in mpl.toolbar\_items) {\n",
629. " var name = mpl.toolbar\_items[toolbar\_ind][0];\n",
630. " var tooltip = mpl.toolbar\_items[toolbar\_ind][1];\n",
631. " var image = mpl.toolbar\_items[toolbar\_ind][2];\n",
632. " var method\_name = mpl.toolbar\_items[toolbar\_ind][3];\n",
633. "\n",
634. " if (!name) {\n",
635. " // put a spacer in here.\n",
636. " continue;\n",
637. " }\n",
638. " var button = $('<button/>');\n",
639. " button.addClass('ui-button ui-widget ui-state-default ui-corner-all ' +\n",
640. " 'ui-button-icon-only');\n",
641. " button.attr('role', 'button');\n",
642. " button.attr('aria-disabled', 'false');\n",
643. " button.click(method\_name, toolbar\_event);\n",
644. " button.mouseover(tooltip, toolbar\_mouse\_event);\n",
645. "\n",
646. " var icon\_img = $('<span/>');\n",
647. " icon\_img.addClass('ui-button-icon-primary ui-icon');\n",
648. " icon\_img.addClass(image);\n",
649. " icon\_img.addClass('ui-corner-all');\n",
650. "\n",
651. " var tooltip\_span = $('<span/>');\n",
652. " tooltip\_span.addClass('ui-button-text');\n",
653. " tooltip\_span.html(tooltip);\n",
654. "\n",
655. " button.append(icon\_img);\n",
656. " button.append(tooltip\_span);\n",
657. "\n",
658. " nav\_element.append(button);\n",
659. " }\n",
660. "\n",
661. " var fmt\_picker\_span = $('<span/>');\n",
662. "\n",
663. " var fmt\_picker = $('<select/>');\n",
664. " fmt\_picker.addClass('mpl-toolbar-option ui-widget ui-widget-content');\n",
665. " fmt\_picker\_span.append(fmt\_picker);\n",
666. " nav\_element.append(fmt\_picker\_span);\n",
667. " this.format\_dropdown = fmt\_picker[0];\n",
668. "\n",
669. " for (var ind in mpl.extensions) {\n",
670. " var fmt = mpl.extensions[ind];\n",
671. " var option = $(\n",
672. " '<option/>', {selected: fmt === mpl.default\_extension}).html(fmt);\n",
673. " fmt\_picker.append(option)\n",
674. " }\n",
675. "\n",
676. " // Add hover states to the ui-buttons\n",
677. " $( \".ui-button\" ).hover(\n",
678. " function() { $(this).addClass(\"ui-state-hover\");},\n",
679. " function() { $(this).removeClass(\"ui-state-hover\");}\n",
680. " );\n",
681. "\n",
682. " var status\_bar = $('<span class=\"mpl-message\"/>');\n",
683. " nav\_element.append(status\_bar);\n",
684. " this.message = status\_bar[0];\n",
685. "}\n",
686. "\n",
687. "mpl.figure.prototype.request\_resize = function(x\_pixels, y\_pixels) {\n",
688. " // Request matplotlib to resize the figure. Matplotlib will then trigger a resize in the client,\n",
689. " // which will in turn request a refresh of the image.\n",
690. " this.send\_message('resize', {'width': x\_pixels, 'height': y\_pixels});\n",
691. "}\n",
692. "\n",
693. "mpl.figure.prototype.send\_message = function(type, properties) {\n",
694. " properties['type'] = type;\n",
695. " properties['figure\_id'] = this.id;\n",
696. " this.ws.send(JSON.stringify(properties));\n",
697. "}\n",
698. "\n",
699. "mpl.figure.prototype.send\_draw\_message = function() {\n",
700. " if (!this.waiting) {\n",
701. " this.waiting = true;\n",
702. " this.ws.send(JSON.stringify({type: \"draw\", figure\_id: this.id}));\n",
703. " }\n",
704. "}\n",
705. "\n",
706. "\n",
707. "mpl.figure.prototype.handle\_save = function(fig, msg) {\n",
708. " var format\_dropdown = fig.format\_dropdown;\n",
709. " var format = format\_dropdown.options[format\_dropdown.selectedIndex].value;\n",
710. " fig.ondownload(fig, format);\n",
711. "}\n",
712. "\n",
713. "\n",
714. "mpl.figure.prototype.handle\_resize = function(fig, msg) {\n",
715. " var size = msg['size'];\n",
716. " if (size[0] != fig.canvas.width || size[1] != fig.canvas.height) {\n",
717. " fig.\_resize\_canvas(size[0], size[1]);\n",
718. " fig.send\_message(\"refresh\", {});\n",
719. " };\n",
720. "}\n",
721. "\n",
722. "mpl.figure.prototype.handle\_rubberband = function(fig, msg) {\n",
723. " var x0 = msg['x0'] / mpl.ratio;\n",
724. " var y0 = (fig.canvas.height - msg['y0']) / mpl.ratio;\n",
725. " var x1 = msg['x1'] / mpl.ratio;\n",
726. " var y1 = (fig.canvas.height - msg['y1']) / mpl.ratio;\n",
727. " x0 = Math.floor(x0) + 0.5;\n",
728. " y0 = Math.floor(y0) + 0.5;\n",
729. " x1 = Math.floor(x1) + 0.5;\n",
730. " y1 = Math.floor(y1) + 0.5;\n",
731. " var min\_x = Math.min(x0, x1);\n",
732. " var min\_y = Math.min(y0, y1);\n",
733. " var width = Math.abs(x1 - x0);\n",
734. " var height = Math.abs(y1 - y0);\n",
735. "\n",
736. " fig.rubberband\_context.clearRect(\n",
737. " 0, 0, fig.canvas.width, fig.canvas.height);\n",
738. "\n",
739. " fig.rubberband\_context.strokeRect(min\_x, min\_y, width, height);\n",
740. "}\n",
741. "\n",
742. "mpl.figure.prototype.handle\_figure\_label = function(fig, msg) {\n",
743. " // Updates the figure title.\n",
744. " fig.header.textContent = msg['label'];\n",
745. "}\n",
746. "\n",
747. "mpl.figure.prototype.handle\_cursor = function(fig, msg) {\n",
748. " var cursor = msg['cursor'];\n",
749. " switch(cursor)\n",
750. " {\n",
751. " case 0:\n",
752. " cursor = 'pointer';\n",
753. " break;\n",
754. " case 1:\n",
755. " cursor = 'default';\n",
756. " break;\n",
757. " case 2:\n",
758. " cursor = 'crosshair';\n",
759. " break;\n",
760. " case 3:\n",
761. " cursor = 'move';\n",
762. " break;\n",
763. " }\n",
764. " fig.rubberband\_canvas.style.cursor = cursor;\n",
765. "}\n",
766. "\n",
767. "mpl.figure.prototype.handle\_message = function(fig, msg) {\n",
768. " fig.message.textContent = msg['message'];\n",
769. "}\n",
770. "\n",
771. "mpl.figure.prototype.handle\_draw = function(fig, msg) {\n",
772. " // Request the server to send over a new figure.\n",
773. " fig.send\_draw\_message();\n",
774. "}\n",
775. "\n",
776. "mpl.figure.prototype.handle\_image\_mode = function(fig, msg) {\n",
777. " fig.image\_mode = msg['mode'];\n",
778. "}\n",
779. "\n",
780. "mpl.figure.prototype.updated\_canvas\_event = function() {\n",
781. " // Called whenever the canvas gets updated.\n",
782. " this.send\_message(\"ack\", {});\n",
783. "}\n",
784. "\n",
785. "// A function to construct a web socket function for onmessage handling.\n",
786. "// Called in the figure constructor.\n",
787. "mpl.figure.prototype.\_make\_on\_message\_function = function(fig) {\n",
788. " return function socket\_on\_message(evt) {\n",
789. " if (evt.data instanceof Blob) {\n",
790. " /\* FIXME: We get \"Resource interpreted as Image but\n",
791. " \* transferred with MIME type text/plain:\" errors on\n",
792. " \* Chrome. But how to set the MIME type? It doesn't seem\n",
793. " \* to be part of the websocket stream \*/\n",
794. " evt.data.type = \"image/png\";\n",
795. "\n",
796. " /\* Free the memory for the previous frames \*/\n",
797. " if (fig.imageObj.src) {\n",
798. " (window.URL || window.webkitURL).revokeObjectURL(\n",
799. " fig.imageObj.src);\n",
800. " }\n",
801. "\n",
802. " fig.imageObj.src = (window.URL || window.webkitURL).createObjectURL(\n",
803. " evt.data);\n",
804. " fig.updated\_canvas\_event();\n",
805. " fig.waiting = false;\n",
806. " return;\n",
807. " }\n",
808. " else if (typeof evt.data === 'string' && evt.data.slice(0, 21) == \"data:image/png;base64\") {\n",
809. " fig.imageObj.src = evt.data;\n",
810. " fig.updated\_canvas\_event();\n",
811. " fig.waiting = false;\n",
812. " return;\n",
813. " }\n",
814. "\n",
815. " var msg = JSON.parse(evt.data);\n",
816. " var msg\_type = msg['type'];\n",
817. "\n",
818. " // Call the \"handle\_{type}\" callback, which takes\n",
819. " // the figure and JSON message as its only arguments.\n",
820. " try {\n",
821. " var callback = fig[\"handle\_\" + msg\_type];\n",
822. " } catch (e) {\n",
823. " console.log(\"No handler for the '\" + msg\_type + \"' message type: \", msg);\n",
824. " return;\n",
825. " }\n",
826. "\n",
827. " if (callback) {\n",
828. " try {\n",
829. " // console.log(\"Handling '\" + msg\_type + \"' message: \", msg);\n",
830. " callback(fig, msg);\n",
831. " } catch (e) {\n",
832. " console.log(\"Exception inside the 'handler\_\" + msg\_type + \"' callback:\", e, e.stack, msg);\n",
833. " }\n",
834. " }\n",
835. " };\n",
836. "}\n",
837. "\n",
838. "// from http://stackoverflow.com/questions/1114465/getting-mouse-location-in-canvas\n",
839. "mpl.findpos = function(e) {\n",
840. " //this section is from http://www.quirksmode.org/js/events\_properties.html\n",
841. " var targ;\n",
842. " if (!e)\n",
843. " e = window.event;\n",
844. " if (e.target)\n",
845. " targ = e.target;\n",
846. " else if (e.srcElement)\n",
847. " targ = e.srcElement;\n",
848. " if (targ.nodeType == 3) // defeat Safari bug\n",
849. " targ = targ.parentNode;\n",
850. "\n",
851. " // jQuery normalizes the pageX and pageY\n",
852. " // pageX,Y are the mouse positions relative to the document\n",
853. " // offset() returns the position of the element relative to the document\n",
854. " var x = e.pageX - $(targ).offset().left;\n",
855. " var y = e.pageY - $(targ).offset().top;\n",
856. "\n",
857. " return {\"x\": x, \"y\": y};\n",
858. "};\n",
859. "\n",
860. "/\*\n",
861. " \* return a copy of an object with only non-object keys\n",
862. " \* we need this to avoid circular references\n",
863. " \* http://stackoverflow.com/a/24161582/3208463\n",
864. " \*/\n",
865. "function simpleKeys (original) {\n",
866. " return Object.keys(original).reduce(function (obj, key) {\n",
867. " if (typeof original[key] !== 'object')\n",
868. " obj[key] = original[key]\n",
869. " return obj;\n",
870. " }, {});\n",
871. "}\n",
872. "\n",
873. "mpl.figure.prototype.mouse\_event = function(event, name) {\n",
874. " var canvas\_pos = mpl.findpos(event)\n",
875. "\n",
876. " if (name === 'button\_press')\n",
877. " {\n",
878. " this.canvas.focus();\n",
879. " this.canvas\_div.focus();\n",
880. " }\n",
881. "\n",
882. " var x = canvas\_pos.x \* mpl.ratio;\n",
883. " var y = canvas\_pos.y \* mpl.ratio;\n",
884. "\n",
885. " this.send\_message(name, {x: x, y: y, button: event.button,\n",
886. " step: event.step,\n",
887. " guiEvent: simpleKeys(event)});\n",
888. "\n",
889. " /\* This prevents the web browser from automatically changing to\n",
890. " \* the text insertion cursor when the button is pressed. We want\n",
891. " \* to control all of the cursor setting manually through the\n",
892. " \* 'cursor' event from matplotlib \*/\n",
893. " event.preventDefault();\n",
894. " return false;\n",
895. "}\n",
896. "\n",
897. "mpl.figure.prototype.\_key\_event\_extra = function(event, name) {\n",
898. " // Handle any extra behaviour associated with a key event\n",
899. "}\n",
900. "\n",
901. "mpl.figure.prototype.key\_event = function(event, name) {\n",
902. "\n",
903. " // Prevent repeat events\n",
904. " if (name == 'key\_press')\n",
905. " {\n",
906. " if (event.which === this.\_key)\n",
907. " return;\n",
908. " else\n",
909. " this.\_key = event.which;\n",
910. " }\n",
911. " if (name == 'key\_release')\n",
912. " this.\_key = null;\n",
913. "\n",
914. " var value = '';\n",
915. " if (event.ctrlKey && event.which != 17)\n",
916. " value += \"ctrl+\";\n",
917. " if (event.altKey && event.which != 18)\n",
918. " value += \"alt+\";\n",
919. " if (event.shiftKey && event.which != 16)\n",
920. " value += \"shift+\";\n",
921. "\n",
922. " value += 'k';\n",
923. " value += event.which.toString();\n",
924. "\n",
925. " this.\_key\_event\_extra(event, name);\n",
926. "\n",
927. " this.send\_message(name, {key: value,\n",
928. " guiEvent: simpleKeys(event)});\n",
929. " return false;\n",
930. "}\n",
931. "\n",
932. "mpl.figure.prototype.toolbar\_button\_onclick = function(name) {\n",
933. " if (name == 'download') {\n",
934. " this.handle\_save(this, null);\n",
935. " } else {\n",
936. " this.send\_message(\"toolbar\_button\", {name: name});\n",
937. " }\n",
938. "};\n",
939. "\n",
940. "mpl.figure.prototype.toolbar\_button\_onmouseover = function(tooltip) {\n",
941. " this.message.textContent = tooltip;\n",
942. "};\n",
943. "mpl.toolbar\_items = [[\"Home\", \"Reset original view\", \"fa fa-home icon-home\", \"home\"], [\"Back\", \"Back to previous view\", \"fa fa-arrow-left icon-arrow-left\", \"back\"], [\"Forward\", \"Forward to next view\", \"fa fa-arrow-right icon-arrow-right\", \"forward\"], [\"\", \"\", \"\", \"\"], [\"Pan\", \"Pan axes with left mouse, zoom with right\", \"fa fa-arrows icon-move\", \"pan\"], [\"Zoom\", \"Zoom to rectangle\", \"fa fa-square-o icon-check-empty\", \"zoom\"], [\"\", \"\", \"\", \"\"], [\"Download\", \"Download plot\", \"fa fa-floppy-o icon-save\", \"download\"]];\n",
944. "\n",
945. "mpl.extensions = [\"eps\", \"jpeg\", \"pdf\", \"png\", \"ps\", \"raw\", \"svg\", \"tif\"];\n",
946. "\n",
947. "mpl.default\_extension = \"png\";var comm\_websocket\_adapter = function(comm) {\n",
948. " // Create a \"websocket\"-like object which calls the given IPython comm\n",
949. " // object with the appropriate methods. Currently this is a non binary\n",
950. " // socket, so there is still some room for performance tuning.\n",
951. " var ws = {};\n",
952. "\n",
953. " ws.close = function() {\n",
954. " comm.close()\n",
955. " };\n",
956. " ws.send = function(m) {\n",
957. " //console.log('sending', m);\n",
958. " comm.send(m);\n",
959. " };\n",
960. " // Register the callback with on\_msg.\n",
961. " comm.on\_msg(function(msg) {\n",
962. " //console.log('receiving', msg['content']['data'], msg);\n",
963. " // Pass the mpl event to the overriden (by mpl) onmessage function.\n",
964. " ws.onmessage(msg['content']['data'])\n",
965. " });\n",
966. " return ws;\n",
967. "}\n",
968. "\n",
969. "mpl.mpl\_figure\_comm = function(comm, msg) {\n",
970. " // This is the function which gets called when the mpl process\n",
971. " // starts-up an IPython Comm through the \"matplotlib\" channel.\n",
972. "\n",
973. " var id = msg.content.data.id;\n",
974. " // Get hold of the div created by the display call when the Comm\n",
975. " // socket was opened in Python.\n",
976. " var element = $(\"#\" + id);\n",
977. " var ws\_proxy = comm\_websocket\_adapter(comm)\n",
978. "\n",
979. " function ondownload(figure, format) {\n",
980. " window.open(figure.imageObj.src);\n",
981. " }\n",
982. "\n",
983. " var fig = new mpl.figure(id, ws\_proxy,\n",
984. " ondownload,\n",
985. " element.get(0));\n",
986. "\n",
987. " // Call onopen now - mpl needs it, as it is assuming we've passed it a real\n",
988. " // web socket which is closed, not our websocket->open comm proxy.\n",
989. " ws\_proxy.onopen();\n",
990. "\n",
991. " fig.parent\_element = element.get(0);\n",
992. " fig.cell\_info = mpl.find\_output\_cell(\"<div id='\" + id + \"'></div>\");\n",
993. " if (!fig.cell\_info) {\n",
994. " console.error(\"Failed to find cell for figure\", id, fig);\n",
995. " return;\n",
996. " }\n",
997. "\n",
998. " var output\_index = fig.cell\_info[2]\n",
999. " var cell = fig.cell\_info[0];\n",
1000. "\n",
1001. "};\n",
1002. "\n",
1003. "mpl.figure.prototype.handle\_close = function(fig, msg) {\n",
1004. " var width = fig.canvas.width/mpl.ratio\n",
1005. " fig.root.unbind('remove')\n",
1006. "\n",
1007. " // Update the output cell to use the data from the current canvas.\n",
1008. " fig.push\_to\_output();\n",
1009. " var dataURL = fig.canvas.toDataURL();\n",
1010. " // Re-enable the keyboard manager in IPython - without this line, in FF,\n",
1011. " // the notebook keyboard shortcuts fail.\n",
1012. " IPython.keyboard\_manager.enable()\n",
1013. " $(fig.parent\_element).html('<img src=\"' + dataURL + '\" width=\"' + width + '\">');\n",
1014. " fig.close\_ws(fig, msg);\n",
1015. "}\n",
1016. "\n",
1017. "mpl.figure.prototype.close\_ws = function(fig, msg){\n",
1018. " fig.send\_message('closing', msg);\n",
1019. " // fig.ws.close()\n",
1020. "}\n",
1021. "\n",
1022. "mpl.figure.prototype.push\_to\_output = function(remove\_interactive) {\n",
1023. " // Turn the data on the canvas into data in the output cell.\n",
1024. " var width = this.canvas.width/mpl.ratio\n",
1025. " var dataURL = this.canvas.toDataURL();\n",
1026. " this.cell\_info[1]['text/html'] = '<img src=\"' + dataURL + '\" width=\"' + width + '\">';\n",
1027. "}\n",
1028. "\n",
1029. "mpl.figure.prototype.updated\_canvas\_event = function() {\n",
1030. " // Tell IPython that the notebook contents must change.\n",
1031. " IPython.notebook.set\_dirty(true);\n",
1032. " this.send\_message(\"ack\", {});\n",
1033. " var fig = this;\n",
1034. " // Wait a second, then push the new image to the DOM so\n",
1035. " // that it is saved nicely (might be nice to debounce this).\n",
1036. " setTimeout(function () { fig.push\_to\_output() }, 1000);\n",
1037. "}\n",
1038. "\n",
1039. "mpl.figure.prototype.\_init\_toolbar = function() {\n",
1040. " var fig = this;\n",
1041. "\n",
1042. " var nav\_element = $('<div/>')\n",
1043. " nav\_element.attr('style', 'width: 100%');\n",
1044. " this.root.append(nav\_element);\n",
1045. "\n",
1046. " // Define a callback function for later on.\n",
1047. " function toolbar\_event(event) {\n",
1048. " return fig.toolbar\_button\_onclick(event['data']);\n",
1049. " }\n",
1050. " function toolbar\_mouse\_event(event) {\n",
1051. " return fig.toolbar\_button\_onmouseover(event['data']);\n",
1052. " }\n",
1053. "\n",
1054. " for(var toolbar\_ind in mpl.toolbar\_items){\n",
1055. " var name = mpl.toolbar\_items[toolbar\_ind][0];\n",
1056. " var tooltip = mpl.toolbar\_items[toolbar\_ind][1];\n",
1057. " var image = mpl.toolbar\_items[toolbar\_ind][2];\n",
1058. " var method\_name = mpl.toolbar\_items[toolbar\_ind][3];\n",
1059. "\n",
1060. " if (!name) { continue; };\n",
1061. "\n",
1062. " var button = $('<button class=\"btn btn-default\" href=\"#\" title=\"' + name + '\"><i class=\"fa ' + image + ' fa-lg\"></i></button>');\n",
1063. " button.click(method\_name, toolbar\_event);\n",
1064. " button.mouseover(tooltip, toolbar\_mouse\_event);\n",
1065. " nav\_element.append(button);\n",
1066. " }\n",
1067. "\n",
1068. " // Add the status bar.\n",
1069. " var status\_bar = $('<span class=\"mpl-message\" style=\"text-align:right; float: right;\"/>');\n",
1070. " nav\_element.append(status\_bar);\n",
1071. " this.message = status\_bar[0];\n",
1072. "\n",
1073. " // Add the close button to the window.\n",
1074. " var buttongrp = $('<div class=\"btn-group inline pull-right\"></div>');\n",
1075. " var button = $('<button class=\"btn btn-mini btn-primary\" href=\"#\" title=\"Stop Interaction\"><i class=\"fa fa-power-off icon-remove icon-large\"></i></button>');\n",
1076. " button.click(function (evt) { fig.handle\_close(fig, {}); } );\n",
1077. " button.mouseover('Stop Interaction', toolbar\_mouse\_event);\n",
1078. " buttongrp.append(button);\n",
1079. " var titlebar = this.root.find($('.ui-dialog-titlebar'));\n",
1080. " titlebar.prepend(buttongrp);\n",
1081. "}\n",
1082. "\n",
1083. "mpl.figure.prototype.\_root\_extra\_style = function(el){\n",
1084. " var fig = this\n",
1085. " el.on(\"remove\", function(){\n",
1086. "\tfig.close\_ws(fig, {});\n",
1087. " });\n",
1088. "}\n",
1089. "\n",
1090. "mpl.figure.prototype.\_canvas\_extra\_style = function(el){\n",
1091. " // this is important to make the div 'focusable\n",
1092. " el.attr('tabindex', 0)\n",
1093. " // reach out to IPython and tell the keyboard manager to turn it's self\n",
1094. " // off when our div gets focus\n",
1095. "\n",
1096. " // location in version 3\n",
1097. " if (IPython.notebook.keyboard\_manager) {\n",
1098. " IPython.notebook.keyboard\_manager.register\_events(el);\n",
1099. " }\n",
1100. " else {\n",
1101. " // location in version 2\n",
1102. " IPython.keyboard\_manager.register\_events(el);\n",
1103. " }\n",
1104. "\n",
1105. "}\n",
1106. "\n",
1107. "mpl.figure.prototype.\_key\_event\_extra = function(event, name) {\n",
1108. " var manager = IPython.notebook.keyboard\_manager;\n",
1109. " if (!manager)\n",
1110. " manager = IPython.keyboard\_manager;\n",
1111. "\n",
1112. " // Check for shift+enter\n",
1113. " if (event.shiftKey && event.which == 13) {\n",
1114. " this.canvas\_div.blur();\n",
1115. " // select the cell after this one\n",
1116. " var index = IPython.notebook.find\_cell\_index(this.cell\_info[0]);\n",
1117. " IPython.notebook.select(index + 1);\n",
1118. " }\n",
1119. "}\n",
1120. "\n",
1121. "mpl.figure.prototype.handle\_save = function(fig, msg) {\n",
1122. " fig.ondownload(fig, null);\n",
1123. "}\n",
1124. "\n",
1125. "\n",
1126. "mpl.find\_output\_cell = function(html\_output) {\n",
1127. " // Return the cell and output element which can be found \*uniquely\* in the notebook.\n",
1128. " // Note - this is a bit hacky, but it is done because the \"notebook\_saving.Notebook\"\n",
1129. " // IPython event is triggered only after the cells have been serialised, which for\n",
1130. " // our purposes (turning an active figure into a static one), is too late.\n",
1131. " var cells = IPython.notebook.get\_cells();\n",
1132. " var ncells = cells.length;\n",
1133. " for (var i=0; i<ncells; i++) {\n",
1134. " var cell = cells[i];\n",
1135. " if (cell.cell\_type === 'code'){\n",
1136. " for (var j=0; j<cell.output\_area.outputs.length; j++) {\n",
1137. " var data = cell.output\_area.outputs[j];\n",
1138. " if (data.data) {\n",
1139. " // IPython >= 3 moved mimebundle to data attribute of output\n",
1140. " data = data.data;\n",
1141. " }\n",
1142. " if (data['text/html'] == html\_output) {\n",
1143. " return [cell, data, j];\n",
1144. " }\n",
1145. " }\n",
1146. " }\n",
1147. " }\n",
1148. "}\n",
1149. "\n",
1150. "// Register the function which deals with the matplotlib target/channel.\n",
1151. "// The kernel may be null if the page has been refreshed.\n",
1152. "if (IPython.notebook.kernel != null) {\n",
1153. " IPython.notebook.kernel.comm\_manager.register\_target('matplotlib', mpl.mpl\_figure\_comm);\n",
1154. "}\n"
1155. ],
1156. "text/plain": [
1157. "<IPython.core.display.Javascript object>"
1158. ]
1159. },
1160. "metadata": {},
1161. "output\_type": "display\_data"
1162. },
1163. {
1164. "data": {
1165. "text/html": [
1166. "<img src=\"data:image/png;base64,\" width=\"640\">"
1167. ],
1168. "text/plain": [
1169. "<IPython.core.display.HTML object>"
1170. ]
1171. },
1172. "metadata": {},
1173. "output\_type": "display\_data"
1174. },
1175. {
1176. "data": {
1177. "text/plain": [
1178. "<matplotlib.collections.PolyCollection at 0x7f6b4e0ee8d0>"
1179. ]
1180. },
1181. "execution\_count": 26,
1182. "metadata": {},
1183. "output\_type": "execute\_result"
1184. }
1185. ],
1186. "source": [
1187. "## plot \n",
1188. "plt.figure()\n",
1189. "plt.plot(LFPR.loc['Global\_FM'], '-', color = 'gray', label = 'Global Level of Femal')\n",
1190. "plt.plot(LFPR.loc['Global\_MA'], '--', color = 'gray', label = 'Global Level of Male')\n",
1191. "\n",
1192. "plt.plot(LFPR.loc['US\_FM'], '-', color = 'darkred', label = 'US Level of Female')\n",
1193. "plt.plot(LFPR.loc['US\_MA'], '--', color = 'darkblue', label = 'US Level of Male')\n",
1194. "\n",
1195. "plt.gca().fill\_between(range(1990, 2016),\n",
1196. " LFPR.loc['US\_FM'], LFPR.loc['US\_MA'],\n",
1197. " facecolor = 'lightblue', alpha = 0.2)"
1198. ]
1199. },
1200. {
1201. "cell\_type": "code",
1202. "execution\_count": 27,
1203. "metadata": {
1204. "collapsed": true
1205. },
1206. "outputs": [],
1207. "source": [
1208. "plt.xlabel('Year')\n",
1209. "plt.ylabel('Labor Force Participation Rate(%)')\n",
1210. "plt.title('Labor Force Participation Rate Trends, Female/Male, United States, Global')\n",
1211. "plt.legend(loc = 1, frameon = False, fontsize = 6)\n",
1212. "plt.gca().spines['top'].set\_visible(False)\n",
1213. "plt.gca().spines['right'].set\_visible(False)"
1214. ]
1215. },
1216. {
1217. "cell\_type": "code",
1218. "execution\_count": 29,
1219. "metadata": {
1220. "collapsed": false,
1221. "scrolled": true
1222. },
1223. "outputs": [],
1224. "source": [
1225. "plt.savefig(\"Labor Force Participation Rate of Female and Male.jpg\") "
1226. ]
1227. },
1228. {
1229. "cell\_type": "code",
1230. "execution\_count": null,
1231. "metadata": {
1232. "collapsed": true
1233. },
1234. "outputs": [],
1235. "source": []
1236. }
1237. ],
1238. "metadata": {
1239. "kernelspec": {
1240. "display\_name": "Python 3",
1241. "language": "python",
1242. "name": "python3"
1243. },
1244. "language\_info": {
1245. "codemirror\_mode": {
1246. "name": "ipython",
1247. "version": 3
1248. },
1249. "file\_extension": ".py",
1250. "mimetype": "text/x-python",
1251. "name": "python",
1252. "nbconvert\_exporter": "python",
1253. "pygments\_lexer": "ipython3",
1254. "version": "3.5.2"
1255. }
1256. },
1257. "nbformat": 4,
1258. "nbformat\_minor": 1
1259. }