Request: for non-interactive backends make fig.canvas.draw() force the render

Link to issue:

https://github.com/matplotlib/matplotlib/issues/16558

Reading the issue, it is clear that the problem lies in the fact that that **canvas.draw()** does not work. Looking at the source code, it appears that **FigureCanvasPgf** does not actually implement the **draw()** method defined in its parent (in **FigureCanvasBase** it is defined but does nothing). Other backends are similar, such as **FigureCanvasPdf** which defines the method but it does not do anything.

Plan:

Figure already has a working **draw()** method which requires a **Renderer** to be supplied to it. The backends already have a reference to the **Figure** instance, so all they would need to be drawn is a **Renderer** object. For some backends, this is trivial, like with the **FigureCanvasPgf**. For others, it is not so simple. For example, **FigureCanvasPdf** does not even own a **Renderer** field. Instead, a new one is created and then used by the **figure.draw()** each time **print_pdf()** is called on it. Fixing this issue would require looking at each of the backends that does not implement **draw()**, and either using the renderer, or finding where a line such as **self.figure.draw(...)** is used, and having the **draw()** method invoke this method.
An important note here is that we do not actually want to invoke existing methods for some of these backends as is, for example: we do not want to use **print_pdf()** in the draw, as that would actually save a file. Instead, we would need to come up with a method to create a renderer in the same way as it is used, have it be used in the **print_pdf()** method, and then also use the same method for our **draw()**. This is because we do not want unintended side-effects from our **draw()**.