1 Overview

Org-mode and matlab-mode provide an efficient and effective system for creating scientific documents which contain MATLAB code and results intermixed with natural language descriptions. You use org babel to execute source code within org-mode files and insert the results back into the org-mode file. You place source code in code blocks:

```
#+begin_src LANGUAGE <HEADER_OPTIONS>
  <CODE>
#+end_src
```

If you type `C-c C-c` in a matlab code block, org-mode will evaluate the code in the `*MATLAB*` matlab-shell buffer and insert the results into the org file, below the code block. See Setup and Export below for instructions on how to setup org babel for matlab code blocks and how to use this example as a template.

2 matlab-code blocks

With org-mode you can embed semantically colored code such as MATLAB within your document and semantically edit it using "Org -> Editing -> Edit Source Example" menu or `C-c '`. For example,

```
x = [12, 64, 24];
plotType = 'pie3';

switch plotType
  case 'bar'
    bar(x)
    title('Bar Graph')
  case {'pie', 'pie3'}
    pie3(x)
    title('Pie Chart')
  otherwise
    warning('Unexpected plot type. No plot created.')
end
```

2.1 matlab code block evaluation with `output` results

The ":results output" matlab code block header option instructs org to insert the MATLAB command window output. The ":exports both" header option says, keep the MATLAB code and also the results when exporting. If you want to see only the results, leave off the ":exports both" option.

```
disp('The results are:')
a = [1, 2; 3, 4]
b = a * 2
```

```
The results are:

a =

     1     2
     3     4

b =

     2     4
     6     8
```

2.2 org matlab code block evaluation reuses the `*MATLAB*` shell buffer

MATLAB code block evaluation reuses the `*MATLAB*` buffer created by `M-x matlab-shell`. The MATLAB workspace state is maintained between evaluations. To avoid reuse of state, you can clear the MATLAB workspace. You can see the history of the evaluations in the `*MATLAB*` buffer. For example, evaluation of the following in order illustrates how state is maintained between the code blocks.

1. First code block

   ```
   a = 123
   ```

   ```
   a =

     123
   ```

2. Second code block uses the variable `a` created by the first code block.

   ```
   b = a + 1000
   ```

   ```
   b =

           1123
   ```

3. Third code block clears the workspace and references 'b' which is known:

   ```
   clear
   c = b * 2
   ```

   ```
   Unrecognized function or variable 'b'.
   ```

2.3 matlab code block evaluation with `verbatim` results

When the matlab code block header contains ":results verbatim", the value of the MATLAB `ans` variable is saved using `writematrix(ans, orgTmpFile, 'Delimiter', 'tab')` and then the contents of the *orgTmpFile* is inserted under the "#+RESULTS:".

```
a = 2 + 3;
ans = magic(a);
```

17	24	1	8	15
23	5	7	14	16
4	6	13	20	22
10	12	19	21	3
11	18	25	2	9

2.4 matlab code block evaluation with `latex` results

With the Symbolic Math Toolbox, you can produce LaTeX using the header option ":results output latex":

```
m = [4*pi, 3*pi; 2*pi, pi];
result = latex(sym(m));
disp(result)
```

$$\left(\begin{array}{cc} 4\,\pi & 3\,\pi \\ 2\,\pi & \pi \end{array}\right)$$

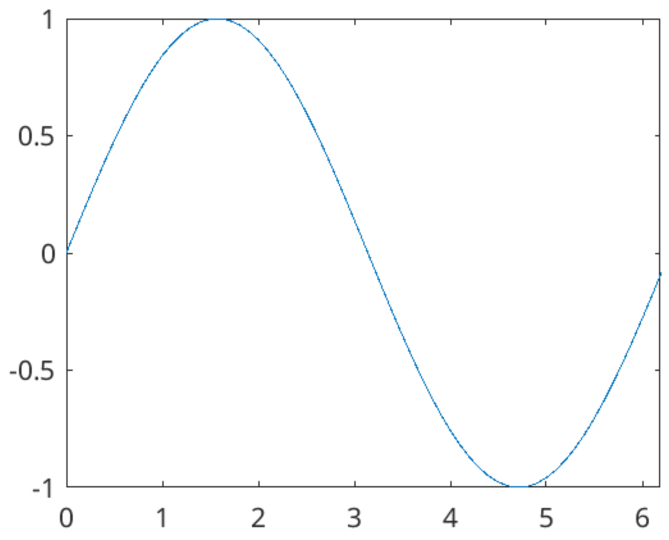
You can use LaTeX directly, for example:

$$\begin{aligned} y(t) &= f_o(t, x_c, x_d, u, P) && -\ \text{outputs} && (1) \\ \dot{x}_c(t) &= f_d(t, x_c, x_d, u, P) && -\ \text{derivatives} && (2) \\ x_d(t+h) &= f_u(t, x_c, x_d, u, P) && -\ \text{update} && (3) \end{aligned}$$

2.5 matlab code block evaluation with `figure` results

You can use org-mode babel evaluate MATLAB code blocks to plot and insert figures back into this file as well as the published (exported) html, LaTeX, pdf, odx (word), etc. file. To do this we use a matlab code block with ":results file graphics" header option. After evaluating the code block, org babel will print the current figure, gcf, using "print -dpng FILE.png" where the name of FILE.png comes from the ":file FILE.png" header option. In this example, we place the ":file FILE.png" header option on a separate line prior to the matlab code block to aid in clarity.

```
t = [0 : 0.1 : 2*pi];
y = sin(t);
plot(t, y);
set(gcf, 'PaperUnits', 'inches', 'PaperPosition', [0 0 4 3]) % Set the size to 4" x 3"
```



3 Setup and Export

1. Enable MATLAB code block export.

   To enable exporting of org containing matlab code blocks, you need to

   `M-x customize-variable RET org-babel-load-languages RET`

   and add matlab, then 'Save for future sessions' using the 'State' button.

   If matlab has not been added to org-babel-load-languages, when you try to evaluate a matlab code block, you will see

   `org-babel-execute-src-block: No org-babel-execute function for matlab!`

2. Use these files as a template for your org files.

   ```
   cd your-working-directory
   cp /path/to/Emacs-MATLAB-Mode/examples/matlab-and-org-mode.org your-file.org
   cp -r /path/to/Emacs-MATLAB-Mode/examples/css .      # If exporting to html
   ```

   Notice that within the *.org file there are several `#+<comments>`. These setup for LaTeX/PDF and HTML export.

3. Configure HTML export.

   You need the htmlize package (1https://melpa.org/#/htmlize) to get coloring for HTML export. For HTML export we set the "#+html_head_extra" properties in our org file to configure CSS.

HTML export uses

- css/styles-from-org.css. This is generated by running

  `M-x org-html-htmlize-generate-css`

  and you'll want to update this for your version of Emacs.
- css/styles.css. This contains customizations which you can edit as desired.

4. Configure PDF export.

   To get colored, better looking PDF, use the minted package. This setup can go in your `~/.emacs`:

```
(defun setup-org-pdf ()
  "Customize org PDF generation for color and more."
  (if (not (boundp 'org-latex-src-block-backend))
      (message "Unable to configure org PDF export because it is too old.")
    (setq org-latex-src-block-backend 'minted
          org-latex-packages-alist '(("cache=false" "minted"))
          org-latex-minted-options '(("xleftmargin" "1em")
                                     ("breaklines" "true")
                                     ("fontsize" "\\small"))
          org-latex-image-default-width ""
          ;; Default value of org-latex-pdf-process does not include -shell-escape which is
          ↪  needed for minted
          ;; Also improve latex log file error messages by adding -file-line-error
          org-latex-pdf-process '("%latex -file-line-error -shell-escape -interaction
          ↪  nonstopmode -output-directory %o %f"
                                  "%latex -file-line-error -shell-escape -interaction
                                  ↪  nonstopmode -output-directory %o %f"
                                  "%latex -file-line-error -shell-escape -interaction
                                  ↪  nonstopmode -output-directory %o %f")
          ;; Keep *.log files to aid in debugging.
          org-latex-logfiles-extensions (remove "log" org-latex-logfiles-extensions))

    ;; Color the hyperlinks, see
    ;;
    ↪  https://tex.stackexchange.com/questions/823/remove-ugly-borders-around-clickable-cross-referenc
    (add-to-list 'org-latex-default-packages-alist

                 ↪  '("colorlinks=true,linkcolor={red!50!black},citecolor={blue!50!black},urlcolor={bl
                  "hyperref" nil))))

(eval-after-load "ox-latex"
  '(setup-org-pdf))
```

5. Export.

   After this setup, you can use the "Org -> Export/Publish" or `C-c C-e` to export to HTML, PDF, etc.