8 The for Loop

\[
\text{for index = expression} \\
\quad \text{Statements to execute} \\
\quad \text{Also called loop body} \\
\text{end}
\]

- *index* can be any variable name
- *expression* gives a vector containing all the values that index will take on, one for each iteration. E.g., \(1:n\).
- The **for** loop is for **definite iteration only**. Do not change the value of *index* within the loop body in hopes of getting out of the loop.

Pattern for doing something \(n\) times

\[
\text{for } i = 1:n \\
\quad \% \text{ do something} \\
\quad \% \ldots \\
\text{end}
\]

9 The while Loop

\[
\text{while expression} \\
\quad \text{statements to execute if expression evaluates to true} \\
\text{end}
\]

Two useful patterns with the **while** Loop

<table>
<thead>
<tr>
<th>Pattern for doing something (n) times</th>
<th>Pattern for doing something an indefinite number of times</th>
</tr>
</thead>
<tbody>
<tr>
<td>(k=1;) | (k\leq n) | (% \text{ do something} | (% \ldots) | (% \text{ initialization} | (% \ldots) | (% \text{ not stopping signal} | (% \text{ do something} | (% \ldots) | (% \text{ update status (variables)} | (% \ldots) | (% \text{ end})</td>
<td></td>
</tr>
</tbody>
</table>

10 Plotting

It is very easy to make plots using MATLAB. An x-y plot can be generated using the built-in function `plot`. The command

\[
\text{plot(a,b,'-', c,d,'*')}\]

will generate a plot with two graphs, one showing the data contained in vectors \(a\) (in x-direction) and \(b\) (in y-direction) as a line and the other showing the data in vectors \(c\) and \(d\) as asterisks. Use the `help` facility in MATLAB to learn more about `plot` and the many formatting options. If you omit the formatting option (\('-'\) and \('*'\) above), the default on most system is to show the data as a line.