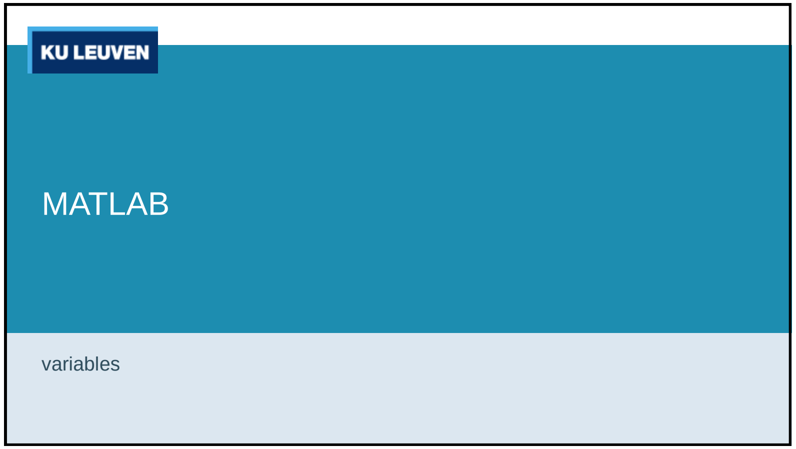
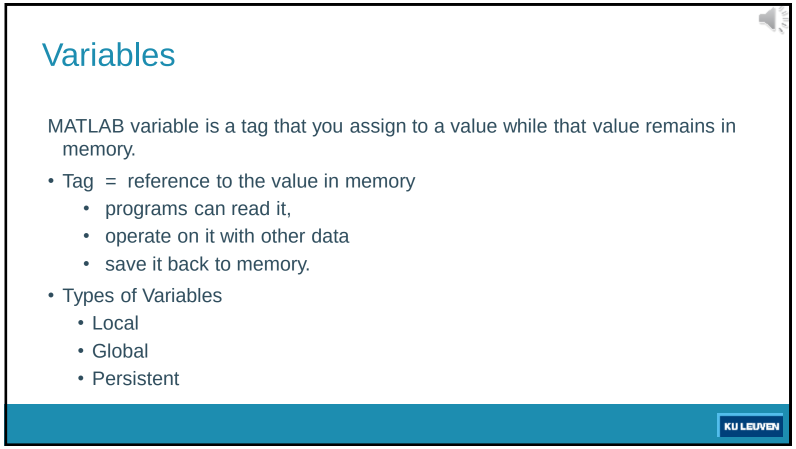


KU LEUVEN

MATLAB

variables

1

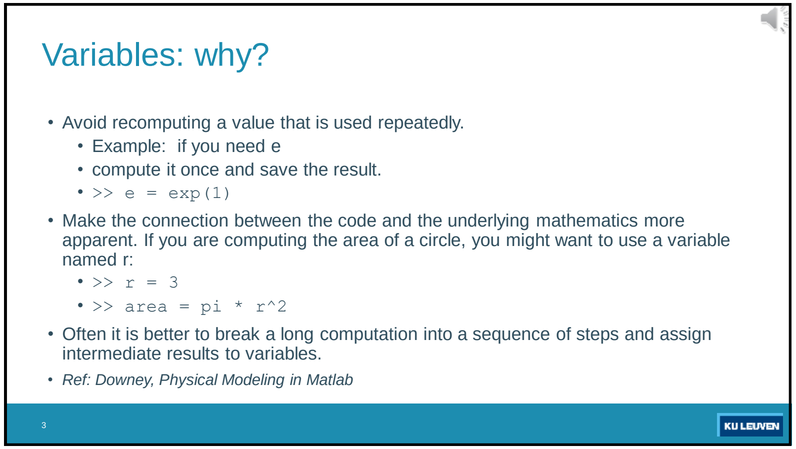


Variables

MATLAB variable is a tag that you assign to a value while that value remains in memory.

- Tag = reference to the value in memory
  - programs can read it,
  - operate on it with other data
  - save it back to memory.
- Types of Variables
  - Local
  - Global
  - Persistent

2

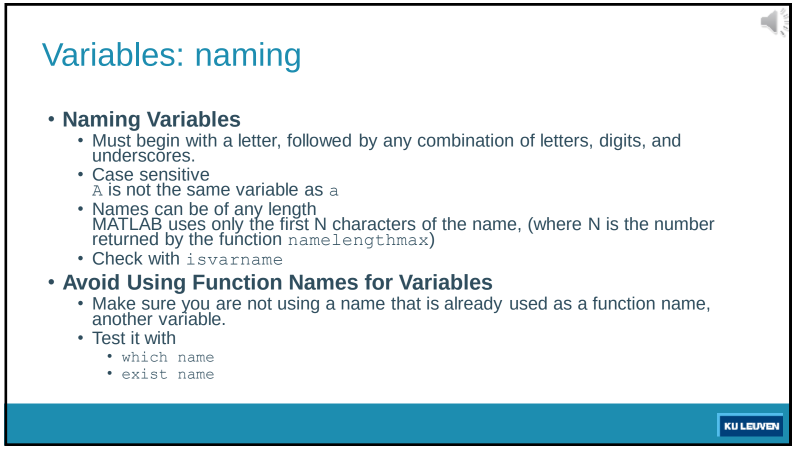


Variables: why?

- Avoid recomputing a value that is used repeatedly.
  - Example: if you need e
  - compute it once and save the result.
  - `>> e = exp(1)`
- Make the connection between the code and the underlying mathematics more apparent. If you are computing the area of a circle, you might want to use a variable named r:
  - `>> r = 3`
  - `>> area = pi * r^2`
- Often it is better to break a long computation into a sequence of steps and assign intermediate results to variables.
- *Ref: Downey, Physical Modeling in Matlab*

3

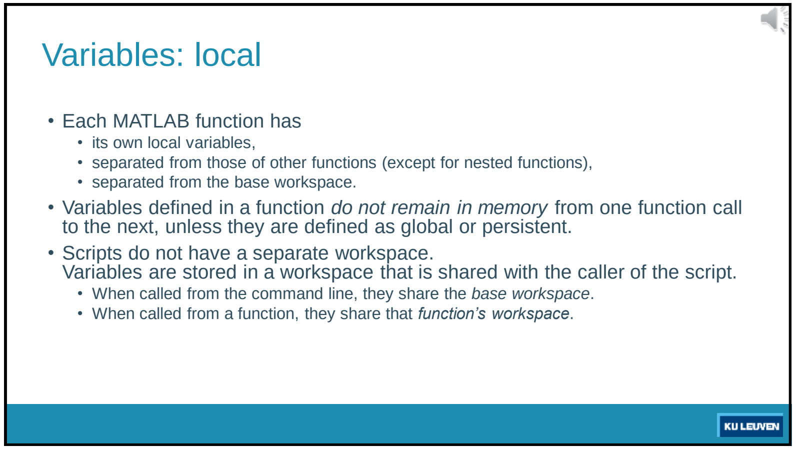
3



Variables: naming

- **Naming Variables**
  - Must begin with a letter, followed by any combination of letters, digits, and underscores.
  - Case sensitive
    `A` is not the same variable as `a`
  - Names can be of any length
    MATLAB uses only the first N characters of the name, (where N is the number returned by the function `namelengthmax`)
  - Check with `isvarname`
- **Avoid Using Function Names for Variables**
  - Make sure you are not using a name that is already used as a function name, another variable.
  - Test it with
    - `which name`
    - `exist name`

4

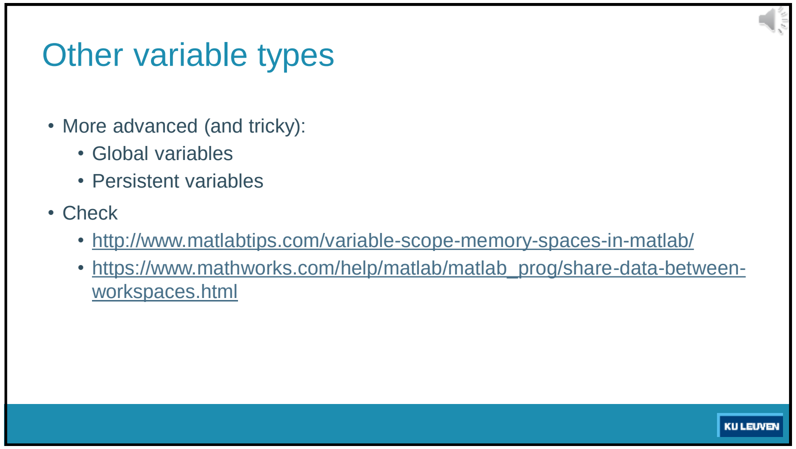


Variables: local

- Each MATLAB function has
  - its own local variables,
  - separated from those of other functions (except for nested functions),
  - separated from the base workspace.
- Variables defined in a function *do not remain in memory* from one function call to the next, unless they are defined as global or persistent.
- Scripts do not have a separate workspace.
  Variables are stored in a workspace that is shared with the caller of the script.
  - When called from the command line, they share the *base workspace*.
  - When called from a function, they share that *function's workspace*.

KU LEUVEN

5

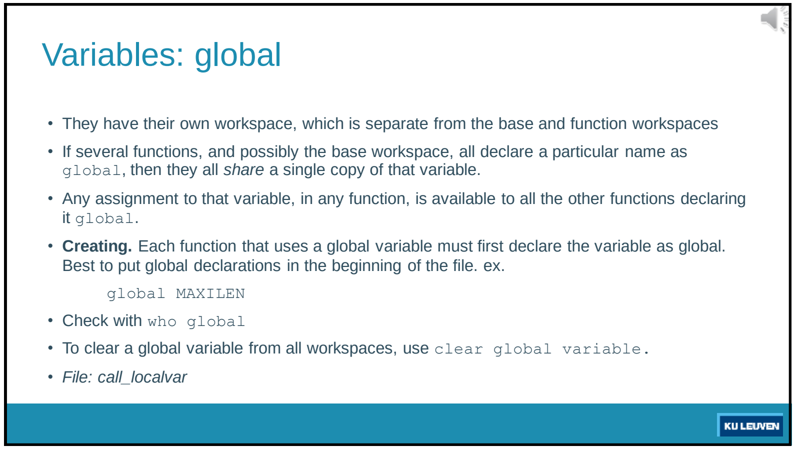


Other variable types

- More advanced (and tricky):
  - Global variables
  - Persistent variables
- Check
  - http://www.matlabtips.com/variable-scope-memory-spaces-in-matlab/
  - https://www.mathworks.com/help/matlab/matlab_prog/share-data-between-workspaces.html

KU LEUVEN

8



Variables: global

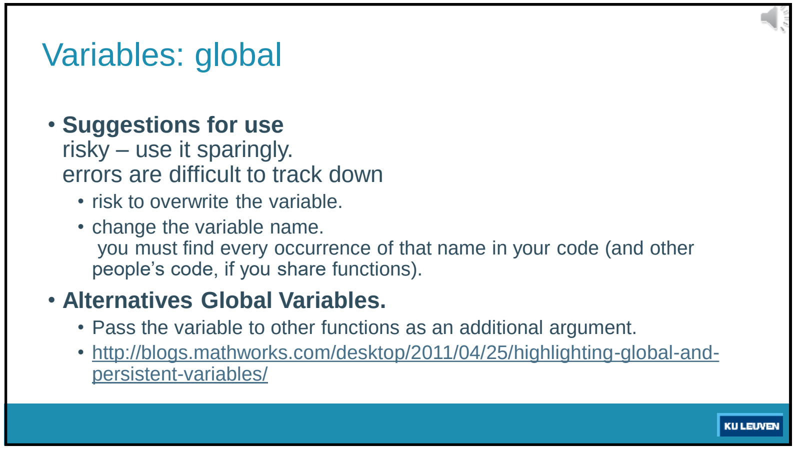
- They have their own workspace, which is separate from the base and function workspaces
- If several functions, and possibly the base workspace, all declare a particular name as `global`, then they all *share* a single copy of that variable.
- Any assignment to that variable, in any function, is available to all the other functions declaring it `global`.
- **Creating.** Each function that uses a global variable must first declare the variable as global. Best to put global declarations in the beginning of the file. ex.

```
    global MAXILEN
```

- Check with `who global`
- To clear a global variable from all workspaces, use `clear global variable`.
- *File: call_localvar*

KU LEUVEN

9



Variables: global

- **Suggestions for use**
  risky – use it sparingly.
  errors are difficult to track down
  - risk to overwrite the variable.
  - change the variable name.
    you must find every occurrence of that name in your code (and other people's code, if you share functions).
- **Alternatives Global Variables.**
  - Pass the variable to other functions as an additional argument.
  - http://blogs.mathworks.com/desktop/2011/04/25/highlighting-global-and-persistent-variables/

KU LEUVEN

10