Effective Programming Practices for Economists

Software engineering

Naming things

Janoś Gabler and Hans-Martin von Gaudecker

Naming is hard but important!

> There are only two hard things in Computer Science: cache invalidation and naming things.
> – Phil Karlton

> You should name a variable using the same care with which you name a first-born child.
> – Robert C. Martin

> One of the miseries of life is that everybody names things a little bit wrong, and so it makes a little bit harder to understand things than it would have been if they had been named differently.
> – Richard Feynman

What happens here?

```
%wage grid
nw = 20;%number of grid points past real wage
wbupper = 4.25;
wblower = 1.25;
wb = log(wblower):(log(wbupper)-log(wblower))/(nw-1):log(wbupper);
wb = exp(wb(:));
n = ny*nd*nw;
q = ones(ny,nd,nw)/(1+rstar); %q(i,j,k)=q(yT_t=y(i),d_t+1=d(j), w_t=wb(k))
qnew = q;
yTix = repmat((1:ny)',[1 nd nw]);%yT = y(yTix);
dix = repmat(1:nd,[ny 1 nw]);

%note now everything is just ny by nw
yTaix = repmat((1:ny)',[1 nw]);
yhat_d0 = -0.35;
yhat_d1 = (1-yhat_d0)/2/max(y);
```

General recommendations

- Avoid abbreviations, especially if ambiguous
  - is `constr` a contsraint or a constructor?
  - is `p` a path or a probability?
  - However, `max` is often better than `maximum`
- Avoid misspelled words
  - `rsnbrck`; use `rosenbrock` instead
  - `lambbda` to avoid the `lambda` keyword; use `lambda_` instead
- Do not use meaningless or hard to see distinctions
  - Do not use `Beta` and `beta` for different concepts

Variable names

- Describe the variable, not what you want to do with it
- Do not append the type to the variable name
  - Bad: `names_list`
  - Good: `names`
- Avoid built in keywords like `list`, `var`, `dict`, `type`
- Never use `n`, `c`, `u` and `s` (they make using debuggers harder!)
- Never use `l` and `I` (they are hard to distinguish)

Function names

- Function names start with a verb in imperative mode
  - Good: `create_`, `calculate_`, `convert_`, `get_`
  - Bad: `return_`, `call_`
- Describe what the function does at a sensible level of abstraction
  - Good: `process_model_specification`
  - Bad: `convert_user_provided_model_dictionary_to_model_class_and_set_defaults`
- If you want to use `and` you need to split your function in two!

The scope rules

- The length of a variable name should be proportional to its scope
  - `i`, `j`, `sr` and `df` are acceptable names if their scope is a few lines
  - they are completely unacceptable if their usage extends over 20 or more lines
- The length of a function name should be **inversely** proportional to it's scope
  - Functions that are used a lot and well known can have short names
    - Example: `minimize` in estimagic
  - Functions that are used internally and not well known should have descriptive names
    - Example: `convert_list_of_dicts_to_dict_of_lists`