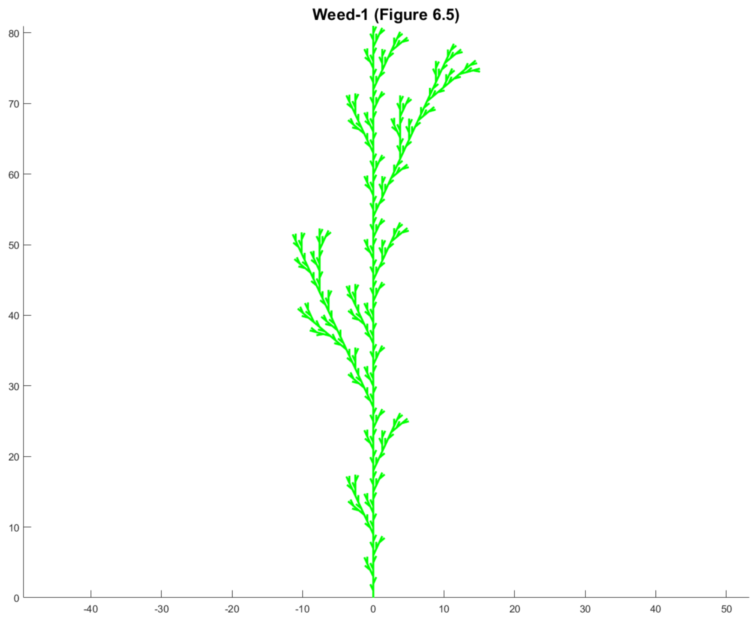
Homework 1.(a)

Modeling Complex Systems, Javier Lobato & Veronica Saz

Due date: Thursday, February 1, 2018

1 Corrections of the code

1. **Name of variables**: the name of the variables has been changed so as that it represents the entity stored.
2. **Degrees to radians**: this change has been done using the function `deg2rad`.
3. **For-loop**: some for loops have been compressed into one single line.
4. **Allocate stack**: initialize the variable stack as an structure in the function `LsysDraw`.
5. **Plot type**: the type of plot has been changed from `line` to `plot`.
6. **Cases**: the cases defined inside the `switch` for 'F', 'G', 'M' and 'N' have been grouped into a single one. The parameters for each letter are chosen from an input structured array using an index that depends on the case.
7. **String to number**: the steps to convert a string into a number will be explained in the following section.
8. **Creation of different functions**: two functions have been created: one function `LsysExpand` that uses the given axiom and rules in the main document (`LsystemDriverA.m`) to obtain a resulting vector and other function called `LsysDraw` to plot the L-system.
9. **Sections**: as the script contains three different cases: *Weed-1, Square-spikes* and *Nature-inspired fractal*, the main file `LsystemDriverA.m` has been divided into three sections that can be evaluated separately.

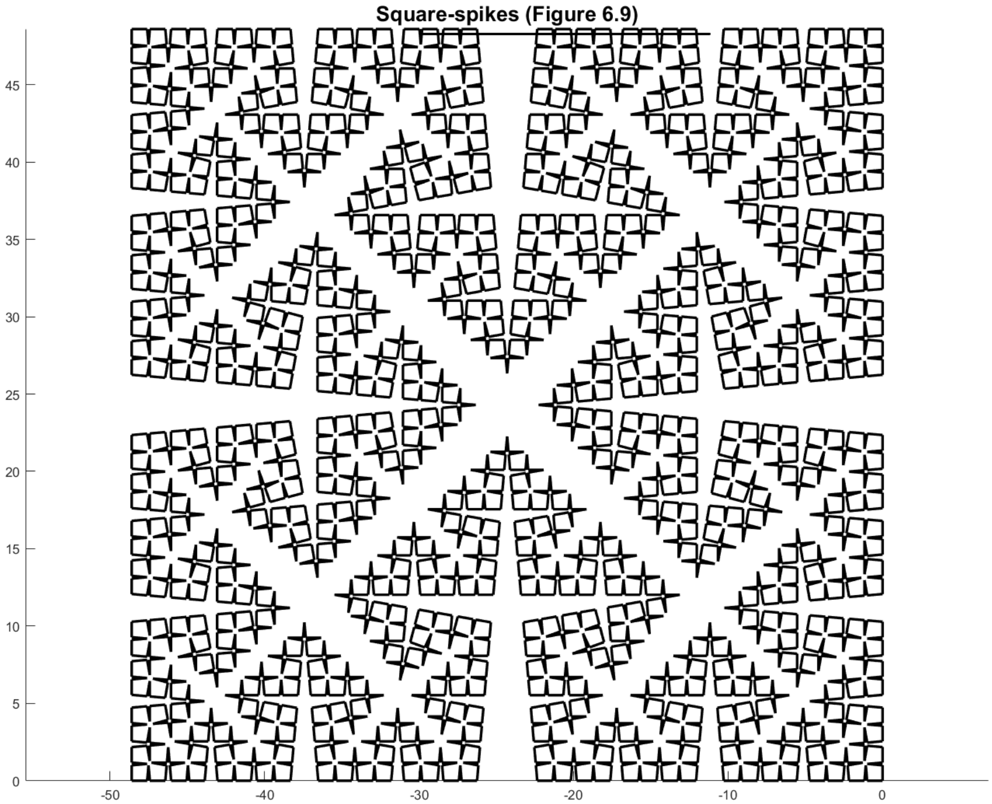


2 Integer multipliers + and -

In order to implement the integer multipliers into the drawing function LsysDraw, the next approach was followed:

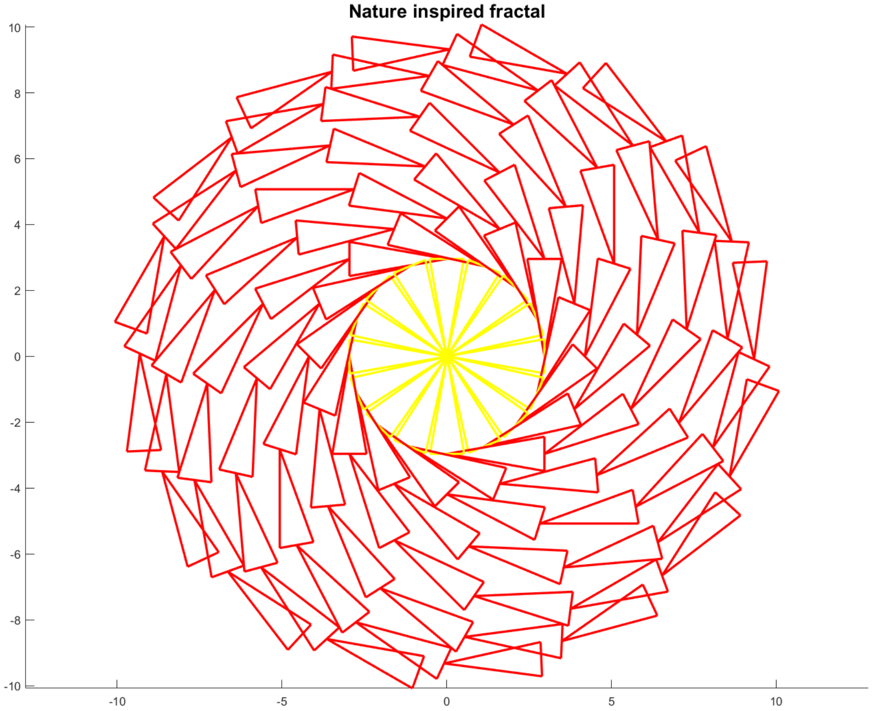
- There is another case in the switch that will differentiate if the string element analyzed is a digit.
- Once inside the case, the digit will be added to a variable called turnNo - initialized with value 0.
- Inside the case, there will be a check on the next element of the string. If it is a number, turnNo will be multiplied by 10, so in the next for-loop iteration, the following digit will be added up to the turnNo.
- If the next element is not a digit, it will be a '+' or a '-'. Inside that case, the number of turnings will be multiplied by the delta angle of each turning. The variable turnNo will be redefined with value 0. In the case that the angle-turning operators exist without a previous integer, the variable value 0 will be changed to 1 to make the turning.

The function LsysExpand did not need any further change to implement the integer multipliers.



3 Nature-inspired fractal

Flowers and plants can be modeled with L-systems, so a *Dahlia* as the one shown in the next figure is a great example that was attempted to be reproduced. The axiom of the L-system consist of the 16 divisions of a circle (X). For each one of the points it will draw a straight petal in yellow (first rule). In addition to drawing that petal, it will include the string Y, that will draw a red turned petal with its origin in the middle of the previous petal (second rule).



4 Listing of the code used

4.1 `LsysExpand.m` function

```matlab
function [ResultingString] = LsysExpand(nReps, axiom, rule)
% LsysExpand: Apply a set of rules of an L-system on the input axiom during
% a specified number of repetitions to give a resulting string
%
%INPUTS:
%nReps = integer value, number of repetitions of the L-system
%axiom = starting seed for the L-system
%rule = set of rules that define the L-system
%
%OUTPUTS:
%ResultingString = modified string after applying the rules to the axiom
%
%Sample test call: ResultingString = LsysExpand(nReps, axiom, rule);
%
%Original code at: http://courses.cit.cornell.edu/bionb441/LSystem/index.html
%Modified by Javier Lobato and Veronica Saz

nRules = length(rule); %get the number of rules from the input set

ResultingString = axiom; %copy the axiom to the resulting string
                         %(for the initialization of the loop)

for i = 1:nReps
    %Convert ResultingString (character vector) to a cell array called RScells
    RScells = cellstr(ResultingString');
    for j = 1:nRules
        %Find occurences of the set of rules in the ResultingString
        hit = strfind(ResultingString, rule(j).before);
        if (length(hit)>=1) %If ocurrences are found
            for k = hit %This will apply each rule to the occurences
                RScells{k} = rule(j).after;
            end
        end
    end
    %Convert RScells from cell array to string
    ResultingString = [RScells{:}]; %No prellocation required
    end
end
```

4.2 LsysDraw.m function

```
function [] = LsysDraw(LsysString, plotParameters, plotTitle, figNo)
% LsysDraw: Draw a string obtained from a L-system with some parameters and
% gives a figure with the specified title. Turtle graphics
%
%INPUTS:
%LsysString = string that contains the result of the function LsysExpand
%plotParameters = structured array with the length and color of each case
%   and the specified delta angle
%plotTitle = string that contains the title of the plot
%figNo = will create different figures to avoid them to overwrite
%
%OUTPUTS:
%No other output than the figure
%
%Sample test call: LsysDraw(LsysString, plotParameters, plotTitle, figNo)
%
%Original code at: http://courses.cit.cornell.edu/bionb441/LSystem/index.html
%Modified by Javier Lobato and Veronica Saz

%Initial state (position and angle) of the turtle
xT = 0;
yT = 0;
aT = 0;

%Convert the specified angle to radians
da = deg2rad(plotParameters(1).delta);

%Init the turtle stack with the required preallocation
stack = struct('xT', cell(length(LsysString), 1), 'yT', cell(length(LsysString), 1), 'aT',
 ↪  cell(length(LsysString), 1));

%Stack counter definition
stckCounter = 1;

%Variable to add on the cummulative turnings (for the cases with digits)
turnNo = 0;

%Create a figure and keep it open until it is completed
figure(figNo)
hold on

for i=1:length(LsysString)
    stringElement = LsysString(i);

    %Different case separation
    switch stringElement

    %Letter case definition
    case {'F', 'G','M','N'}
        %Assign an index for each letter correspondinng to one index of the
        %structured array of the input
        if stringElement == 'F'
            j = 1;
        elseif stringElement == 'G'
```

```
            j = 2;
        elseif stringElement == 'M'
            j = 3;
        elseif stringElement == 'N'
            j = 4;
        end

        newxT = xT + plotParameters(j).length*cos(aT);
        newyT = yT + plotParameters(j).length*sin(aT);
        plot([yT newyT], [xT newxT],'color',plotParameters(j).color, 'linewidth',2);
        xT = newxT;
        yT = newyT;

    case {'X', 'Y'}
        %Do nothing!

    case '+' %Clockwise turning angle
        %In case the number is zero (initialization value) it will be one
        %to make a turning equal to delta
        if turnNo == 0
            turnNo = 1;
        end
        aT = aT + turnNo*da; %Multiply the delta angle times the specified digit number
        turnNo = 0; %Assign the value of turnings to zero

    case '-' %Counterclockwise turning angle
        %In case the number is zero (initialization value) it will be one
        %to make a turning equal to delta
        if turnNo == 0
            turnNo = 1;
        end
        aT = aT - turnNo*da; %Multiply the delta angle times the specified digit number
        turnNo = 0; %Assign the value of turnings to zero

    case '[' %Push the stack with current values
        stack(stckCounter).xT = xT ;
        stack(stckCounter).yT = yT ;
        stack(stckCounter).aT = aT ;
        stckCounter = stckCounter +1 ;

    case ']' %Pop the stack taking the last values
        stckCounter = stckCounter-1 ;
        xT = stack(stckCounter).xT ;
        yT = stack(stckCounter).yT ;
        aT = stack(stckCounter).aT ;

    case {'0','1','2','3','4','5','6','7','8','9'} %Digit case
        %Takes the digit value
        turnNo = turnNo + str2num(LsysString(i));
        %Checks the next element of the string. In case it is another
        %digit, it multiplies the value of turnNo by 10 and it will add the
        %next digit in the following for-loop repetition
        if ~mod(str2num(LsysString(i+1)),1) == 1
            turnNo = turnNo*10;
        end

    otherwise
```

```
            disp('error')
        return
    end
end

hold off

%Plot configuration and title
axis equal
title(plotTitle, 'FontSize',16)

end
```

4.3 LsystemDriverA.m script

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                              HOMEWORK #1                                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%This code was originally downloaded from the following web site
%   http://courses.cit.cornell.edu/bionb441/LSystem/index.html
%
%Given by Margaret Eppstein for the course
%    CSYS 302 'Modeling Complex Systems'
%
%Modified by Javier Lobato and Veronica Saz

%% WEED 1. Figure 6.5
clear all

%Axiom
axiom = 'F';

%Set of rules
rule(1).before = 'F';
rule(1).after = 'F[-F]F[+F]F';

%Number of repetitions
nReps = 4;

%String calculation
ResultingString = LsysExpand(nReps, axiom, rule);

%Plot parameters definition
plotParameters = struct('length', cell(1, 1), 'color', cell(1, 1));

plotParameters(1).length = 1; %Length of case F
plotParameters(1).color = [0.0 1.0 0.0]; %Green
plotParameters(1).delta = 25;

%Turtle graphic plotter
LsysDraw(ResultingString, plotParameters, 'Weed-1 (Figure 6.5)', 1);

%% SQUARE-SPIKES. Figure 6.9
clear all

%Axiom
axiom = 'F18-F18-F18-F';

%Set of rules
rule(1).before = 'F';
rule(1).after = 'F17-F34+F17-F';

%Number of repetitions
nReps = 5;

%String calculation
ResultingString = LsysExpand(nReps, axiom, rule);

```

```
%Plot parameters definition
plotParameters = struct('length', cell(1, 1), 'color', cell(1, 1));

plotParameters(1).length = 1; %Length of case F
plotParameters(1).color = [0.0 0.0 0.0]; %Black
plotParameters(1).delta = 5;

%Turtle graphic plotter
LsysDraw(ResultingString, plotParameters, 'Square-spikes (Figure 6.9)', 2);


%% NATURE INSPIRED FRACTAL
clear all


%Axiom
axiom = 'X9+X9+X9+X9+X9+X9+X9+X9+X9+X9+X9+X9+X9+X9+X9+X9+';

%Set of rules (X and Y will not draw, F-G-M-N will draw)
rule(1).before = 'X';
rule(1).after = '4+F40-GYG40-F68-';

rule(2).before = 'Y';
rule(2).after = '4+M40-N[25+Y]N40-M68-';

%Number of repetitions
nReps = 7;

%String calculation
ResultingString = LsysExpand(nReps, axiom, rule);

%Plot parameters definition
plotParameters = struct('length', cell(4, 1), 'color', cell(4, 1),'delta',cell(1,1));

plotParameters(1).length = 3; %Length of segment F
plotParameters(2).length = 0.52; %Length of segment G
plotParameters(3).length = 3; %Length of segment M
plotParameters(4).length = 0.52; %Length of segment N
plotParameters(1).color = [1.0 1.0 0.0]; %Yellow color (F)
plotParameters(2).color = [1.0 1.0 0.0]; %Yellow color (G)
plotParameters(3).color = [1.0 0.0 0.0]; %Red color (M)
plotParameters(4).color = [1.0 0.0 0.0]; %Red color (N)
plotParameters(1).delta = 2.5;

%Turtle graphic plotter
LsysDraw(ResultingString, plotParameters, 'Nature inspired fractal', 3);
```