warmup Homework - CS520

William Tirone, Marcus Ortiz

January 27, 2024

1. Analysis

1. Vector space & inner product space.

a) Yes - dimension = ∞

b) Yes - dimension = $\binom{d+k}{k}$

c) Yes - dimension = ∞

d) Yes

e) Yes

optional) Yes - d) and e) are just a special case of this.

2. Linear (mapping) operators.

a) Yes

b) No

c) Yes. This is closed under both addition and scalar multiplication, so it satisfies the properties of a linear operator. Depending on the values of A and B, there may or may not be non-zero vectors in the nullspace. If A and B are the identity matrix, then everything is in the nullspace.

d) No

e) Yes - the nullspace has non-zero vectors.

f) Yes, integration is linear. The nullspace has non-zero vectors.

g) Yes, differentiation is linear. The nullspace has non-zero vectors.

h) Yes. The nullspace has non-zero vectors.

i) Yes

3. Basis-specific representations & transformations.

a) Yes

b) Yes

2 MATLAB

1)

The script from ChatGPT ran, and took 0.013858 seconds for 1000 iterations and 0.019636 seconds for 2000 iterations which is, I believe, what the variable `maxsteps` is in `demo bifurcation diagram`. The rendered diagram also looks very similar. I also just timed the loop and initialization and not the time it took the plots to render or the time to generate the plots.

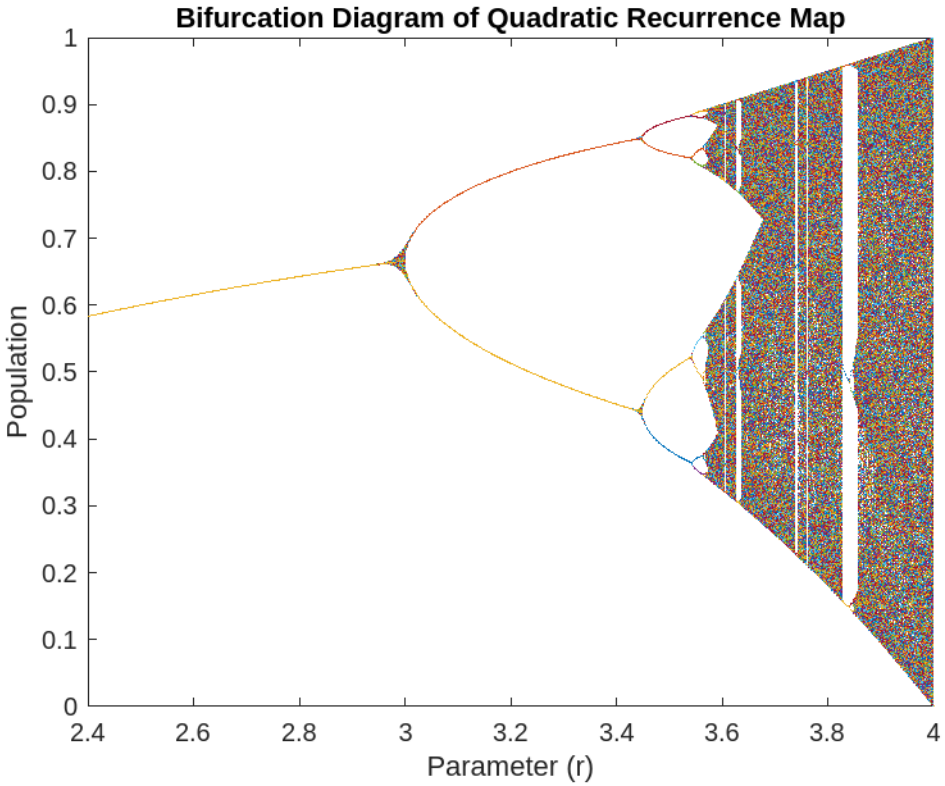

Figure 1: Chat GPT Script Result

2) Differences:

- The initial values are different. For example, the r values from ChatGPT are `linspace(2.4, 4, 1000)` which is a finer resolution than the provided script which stepped through r by 0.01.
- The transient iterations from ChatGPT are set at 100 while the script provided has 300.
- The bifurcation diagram is initialized slightly differently, though it's remarkable how similarly the two scripts are up to this point.
- ChatGPT does not consider `nrs = floor((rmax-rmin)/dr);` though I think it may have handled this with the initial `linspace()` call.
- ChatGPT looped over the parameters with 2 nested `for` loops and an `if` statement. It looks like these accomplish identical tasks, so it's probably just up to personal preference to choose either syntax.

3)

We changed the range to $r \in [-2, 2]$ and the pattern in the data looks very different. So a few values are r = -2, -1.5, 1.

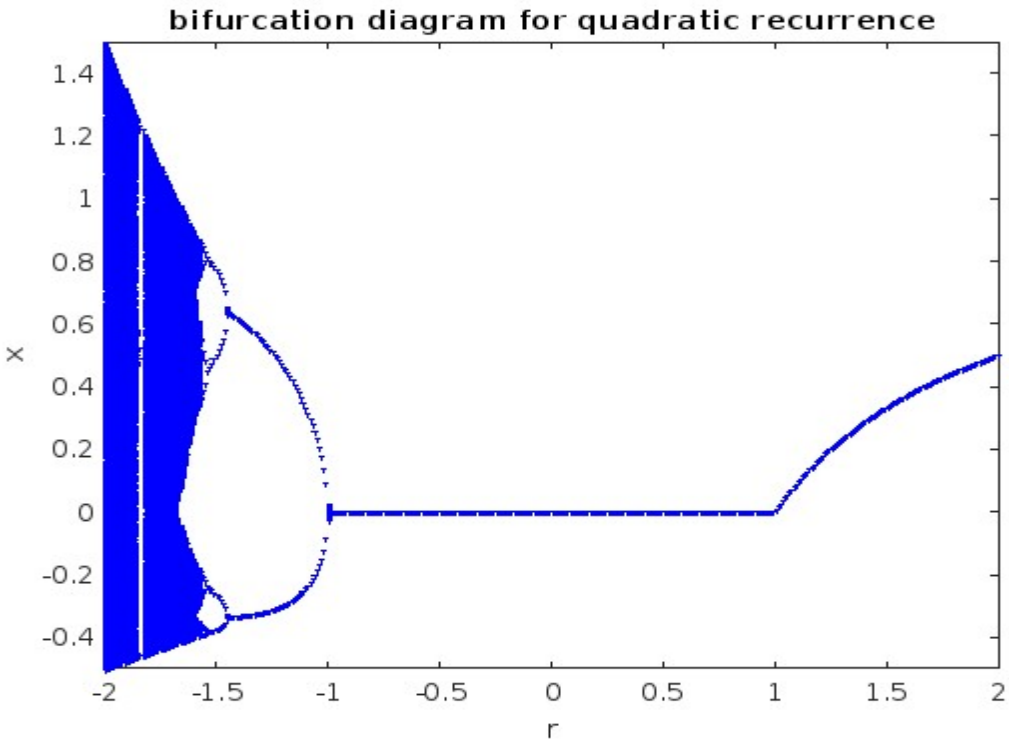

Figure 2: Different r values

optional a)

see matrix_generator.m and simplified.m (also submitted on sakai). Note that simplified.m includes our array data matrix operation, so the call to the matrix function in the script is the array version, but it would work the same either way.

optional b)

see operation_matrix_generator.m (also submitted on sakai). This resulted in a run time of 0.006014 seconds. Compared to our loop matrix creation in matrix_generator.m that ran in 0.00755 seconds, it reduced the run time by about 0.0015 seconds.

optional c)

Recurrence Relation: $x_n = r * x_{n-1} - x_{n-1}^2$
Sequence: $r * x_0 - x_0^2, r^2 x_0 - r x_0^2 - (r * x_0 - x_0^2)^2, ...$