Back to linear recurrences

Example 54. (review) Consider the sequence a_n defined by $a_{n+2} = a_{n+1} + 2a_n$ and $a_0 = 1$, $a_1 = 8$.

(a) Determine the first few terms of the sequence.

(b) Find a Binet-like formula for a_n.

(c) Determine $\lim_{n\to\infty} \frac{a_{n+1}}{a_n}$.

Solution.

(a) $a_2 = 10$, $a_3 = 26$

(b) The recursion can be written as $p(N)a_n = 0$ where $p(N) = N^2 - N - 2$ has roots $2, -1$.
Hence, $a_n = \alpha_1 2^n + \alpha_2 (-1)^n$ and we only need to figure out the two unknowns α_1, α_2. We can do that using the two initial conditions: $a_0 = \alpha_1 + \alpha_2 = 1$, $a_1 = 2\alpha_1 - \alpha_2 = 8$.
Solving, we find $\alpha_1 = 3$ and $\alpha_2 = -2$ so that, in conclusion, $a_n = 3 \cdot 2^n - 2(-1)^n$.

(c) It follows from the Binet-like formula that $\lim_{n\to\infty} \frac{a_{n+1}}{a_n} = 2$.

Systems of recurrence equations

Example 55. Write the (second-order) RE $a_{n+2} = a_{n+1} + 2a_n$, with $a_0 = 1$, $a_1 = 8$, as a system of (first-order) recurrences.

Solution. Write $b_n = a_{n+1}$.

Then, $a_{n+2} = a_{n+1} + 2a_n$ translates into the first-order system $\begin{cases} a_{n+1} = b_n \\ b_{n+1} = 2a_n + b_n \end{cases}$.

Let $\boldsymbol{a}_n = \begin{bmatrix} a_n \\ b_n \end{bmatrix}$. Then, in matrix form, the RE is $\boldsymbol{a}_{n+1} = \begin{bmatrix} 0 & 1 \\ 2 & 1 \end{bmatrix} \boldsymbol{a}_n$, with $\boldsymbol{a}_0 = \begin{bmatrix} 1 \\ 8 \end{bmatrix}$.

Comment. Consequently, $\boldsymbol{a}_n = \begin{bmatrix} 0 & 1 \\ 2 & 1 \end{bmatrix}^n \boldsymbol{a}_0 = \begin{bmatrix} 0 & 1 \\ 2 & 1 \end{bmatrix}^n \begin{bmatrix} 1 \\ 8 \end{bmatrix}$. Solving (systems of) REs is equivalent to computing powers of matrices!

Example 56. Write $a_{n+3} - 4a_{n+2} + a_{n+1} + 6a_n = 0$ as a system of (first-order) recurrences.

Solution. Write $b_n = a_{n+1}$ and $c_n = a_{n+2}$.

Then, $a_{n+3} - 4a_{n+2} + a_{n+1} + 6a_n = 0$ translates into the first-order system $\begin{cases} a_{n+1} = b_n \\ b_{n+1} = c_n \\ c_{n+1} = -6a_n - b_n + 4c_n \end{cases}$.

Let $\boldsymbol{a}_n = \begin{bmatrix} a_n \\ b_n \\ c_n \end{bmatrix}$. Then, in matrix form, the RE is $\boldsymbol{a}_{n+1} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -6 & -1 & 4 \end{bmatrix} \boldsymbol{a}_n$.

Important comment. Consequently, $\boldsymbol{a}_n = M^n \boldsymbol{a}_0$, where M is the matrix above.

(systems of REs) The unique solution to $\boldsymbol{a}_{n+1} = M\boldsymbol{a}_n$, $\boldsymbol{a}_0 = \boldsymbol{c}$ is $\boldsymbol{a}_n = M^n \boldsymbol{c}$.

We will say that M^n is the **fundamental matrix solution** to $\boldsymbol{a}_{n+1} = M\boldsymbol{a}_n$ with $\boldsymbol{a}_0 = I$ (the identity matrix).

Sage

In practice, we are happy to let a machine do tedious computations. Let us see how to use the open-source computer algebra system **Sage** to do basic computations for us.

Sage is freely available at sagemath.org. Instead of installing it locally (it's huge!) we can conveniently use it in the cloud at cocalc.com from any browser.

[For basic computations, you can also simply use the textbox on our course website.]

Sage is built as a **Python** library, so any Python code is valid. For starters, we will use it as a fancy calculator.

Example 57. To solve the differential equation $y'' + 4y' + 4y = 7e^{-2x}$, as we did in Example 30, we can use the following:

```
>>> x = var('x')
>>> y = function('y')(x)
>>> desolve(diff(y,x,2) + 4*diff(y,x) + 4*y == 7*exp(-2*x), y)
```

$$\frac{7}{2} x^2 e^{(-2x)} + (K_2 x + K_1) e^{(-2x)}$$

This confirms, as we had found, that the general solution is $y(x) = \left(C_1 + C_2 x + \frac{7}{2}x^2\right)e^{-2x}$.

Example 58. Similarly, Sage can solve initial value problems such as $y'' - y' - 2y = 0$ with initial conditions $y(0) = 4$, $y'(0) = 5$.

```
>>> x = var('x')
>>> y = function('y')(x)
>>> desolve(diff(y,x,2) - diff(y,x) - 2*y == 0, y, ics=[0,4,5])
```

$$3\, e^{(2x)} + e^{(-x)}$$

This matches the (unique) solution $y(x) = 3e^{2x} + e^{-x}$ that we derived in Example 18.

Armin Straub
straub@southalabama.edu