orbitalsETON

March 29, 2022

1 Continued fraction expansion

This code gives the continued fraction expansion of any number. I.e for $x \in \mathbb{R}$ of the form

$$x = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \dots}}}$$

it gives the $[a_0, a_1, a_2, \dots, a_n]$

[4]:
```
function cfexp(x::Union{Irrational, Float32, Float64}, n::Int64, trace::Bool =␣
 ↪false)
    A = Vector{Int}(undef, n + 1)
    m = 1
    a = floor(x)
    A[1] = a

    while m < n + 1
        a = (x != 0 ? floor(1 / x) : 0)
        x = (x == .0 ? .0 : (1 / x - floor(1 / x)))
        trace ? println(x) :
        m += 1
        A[m] = a ## In julia indexing begins with 1
    end

    return transpose(A)
end
```

[4]:
```
cfexp (generic function with 2 methods)
```

This function basically checks if the result of the first is what it should be.

[5]:
```
function check(z)
    s = 0
    a = length(z)

    while a > 1
        s += z[a]
        s = (s == 0 ? 0 : 1 / s)
        a -= 1
    end

    return(s)
end
```

[5]:
```
check (generic function with 1 method)
```

Notice this function for any $x \in \mathbb{Q}$ the continued fraction expansion is not unique but this function only returns one vector.

Continued fraction expansion for $\frac{1}{2}$ first 3 digits

[6]:
```
z = cfexp(.5, 3)
println(z)
```

```
[0 2 0 0]
```

[7]:
```
check(z) - 1 / 2
```

[7]:
```
0.0
```

For π, julia has an incredible precision so this should be should be somewhat accurate (first 1000 digits)

[8]:
```
@time begin
    z = cfexp(pi, 1000)
    println(z)
end
```

```
[3 0 3 7 15 1 292 1 1 1 2 1 3 1 14 3 3 23 1 1 7 4 35 1 1 1 2 3 3 3 3 1 1 14 6 4
5 1 7 1 5 1 1 3 18 2 1 2 4 2 96 2 3 2 1 1 6 1 6 2 5 64 1 2 3 1 17 5 1 12 3 2 1 1
1 1 2 2 1 4 1 1 2 2 22 1 2 1 6 1 16 1 2 3 2 4 2 5 2 3 1 1 3 2 1 7 6 4 4 3 1 61
20 11 4 1 1 4 3 1 1 3 2 1 2 1 13 2 12 2 1 1 1 1 3 1 1 1 5 10 8 9 4 1 5 1 1 2 4 1
7 3 5 4 66 13 3 1 1 6 32 1 5 4 4 6 1 2 4 1 1 1 1 2 2 1 1 1 7 2 1 2 92 2 1 5 4 2
13 2 1 1 22 2 1 3 4 6 1 22 11 3 1 1 2 2 5 1 14 8 10 3 2 1 5 8 4 7 2 4 2 1 2 1 2
1 1 5 1 3 1 2 2 2 1 1 4 2 14 1 1 6 2 2 1 1 2 1 15 2 3 2 3 53 56 4 2 1 7 1 55 1 2
7 2 9 1 46 2 15 37 7 34 1 2 1 5 1 1 2 2 4 1 2 4 1 1 2 1 9 5 3 3 4 2 6 2 2 2 3 5
1 1 4 2 21 1 1 1 1 1 1 3 1 1 1 3 1 33 1 10 2 1 8 4 3 1 1 1 6 1 1 1 15 1 2 4 264
4 1 2 1 27 1 10 1 23 1 4 7 1 4 2 5 4 4 6 4 2 1 2 8 1 6 6 1 1 6 4 3 1 2 151 1 1
22 1 4 2 2 2 1 72 1 1 6 85 3 1 1 1 3 4 2 3 4 7 3 16 1 1 5 1 1 3 1 1 3 2 2 3 5 24
3 2 1 6 22 1 259 5 4 2 1 1 3 3 13 1 4 1 47 31 1 6 5 95 1 1 1 1 1 1 1 1 4 2 2 5 1
7 1 2 26 1 2 593 27 5 2 7 1 1 1 2 9 1 2 2 1 2 1 1 1 1 2 2 2 1 4 1 1 1 7 3 51 1 3
1 7 1 6 1 1 1 1 1 5 1 1 129 1 6 1 2 2 2 1 2 1 2 31 1 6 30 2 1 87 1 53 2 1 1 2 1
3 2 3 1 28 1 7 1 3 1 1 2 2 8 2 9 1 2 1 1 1 3 1 1 2 2 1 6 3 35 11 1 1 3 1 2 1 2 1
6 1 2 2 5 5 1 3 2 1 1 4 1 1 5 1 1 1 2 1 13 2 680 2 2 3 1 2 4 1 1 3 1 12 5 1 11 1
6 2 1 2 2 28 2 1 3 2 9 1 1 4 3 1 1 4 1 2 1 1 21 4 2 9 2 6 1 12 26 1 2 77 13 1 1
16 5 2 1 3 1 1 1 1 1 34 1 203 78 1 6 1 2 5 3 2 1 3 3 2 14 2 3 1 1 1 5 9 8 15 211
3 1 159 11 1 11 1 8 5 1 2 1 1 3 436 141 3 66 3 1 9 21 1 1 6 1 1 2 2 1 1 1 1 4 28
1 2 1 1 19 1 9 1 6 1 24 17 9 2 10 1 9 17 1 1 1 1 2 11 5 2 20 6 1 1 1 2 5 15 3 2
1 1 7 1 16 1 4 1 8 1 7 1 5 2 5 2 1 9 3 1 1 4 4 1 1 5 4 1 2 1 1 1 19 1 3 1 10 1 2
2 3 1 1 6 1 2 1 5 17 1 12 8 31 11 2 1 8 1 1 2 2 2 2 7 1 1 75 1 32 3 1 151 1 2 87
2 3 2 19 10 2 1 5 1 2 1 2 1 16 3 2 2 10 3 1 1 133 1 2 1 1 1 2 1 1 2 8 6 1 24 1
15 1 1 1 52 1 1 1 4 3 1 11 4 1 15 1 1 14 1 5 50 1 1 4 2 2 1 7 11 5 1 2 2 1 2 5 1
5 2 2 8 1 1 4 9 1 25 1 1 1 1 20 7 1 5 1 3 6 10 289 1 28 1 19 6 1 8 2 1 4 3 3 15
1 2 131 1 6 5 2 18 1 34 4 2 1 61 2 14 43 2 2 1 1 1 1 1 1 1 19 6 1 3 2 1 51 1 5 2
1 10 4 3 1 3 2 3 15 1 4 7 2 1 5 1 1 1 62 1 20 1 6 11 1 3 2 3 24 8 1 2 10 1 1]
  0.650357 seconds (14.74 k allocations: 459.984 KiB)
```

Quite accurate

[9]:

```
check(z) - pi
```

[9]:

```
0.0
```

First 100000 digits (they'll be printed as the last chunk of the notebook):

For $\sqrt{2}$, first 1000 digits:

[10]:

```
@time begin
    z = cfexp(sqrt(2), 1000)
    println(z)
end
```

```
[1 0 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 3 3 1 3 1 1 2 1809 1 2 5 2
2 1 2 1 3 3 4 1 1 3 12 2 2 10 32 2 1 4 2 2 6 1 1 2 8 2 4 1 4 2 1 3 5 7 152 5 4
87 1 1 2 1 2 1 12 1 6 1 1 1 1 1 3 11 1 18 3 1 8 9 1 1 2 1 9 2 8 15 1 2 16 1 4 1
1 1 1 2 1 11 1 1 32 18 1 2 2 1 6 16 2 1 11 1 6 1 1 1 5 1 12 5 13 14 1 1 1 12 1 2
1 1 1 1 1 2 1 6 1 3 3 4 1 1 6 3 3 3 4 4 1 5 1 4 9 2 1 8 17 4 1 1 4 5 1 2 2 1 2 6
1 1 1 1 1 1 1 1 3 157 1 2 26 1 12 1 2 1 4 1 1 15 12 3 8 1 1 1 1 2 1 1 1 1 1 5 1
2 1 1 4 1 10 1 7 3 1 6 1 3 1 1 1 11 1 5 1 1 4 1 1 1 1 2 1 3 2 7 4 16 1 1 2 2 1 2
2 1 2 1 3 13 5 1 9 9 5 1 1 15 1 1 2 42 9 9 1 1 6 24 2 2 1 11 1 3 8 1 8 1 73 2 6
12 7 3 3 12 1 2 1 1 15 4 1 1 1 10 7 5 18 1 3 2 1 1 1 24 15 1 1 1 1 2 5 1 36 1 5
1 1 1 2 1 2 3 2 6 1 4 2 14 1 4 6 1 2 1 1 4 1 3 4 1 2 2 2 1 2 1 4 1 8 3 15 12 1 1
1 1 1 8 2 3 5 1 1 3 1 12 1 127 1 3 1 21 7 1 1 7 1 1 12 5 2 1 5 15 2 1 4 1 1 90 2
2 1 10 1 14 6 1 1 1 2 2 35 2 1 1 1 2 1 6 1 1 3 2 9 1 118 2 1 47 2 5 4 2 2 5 1 3
1 1 1 1 2 4 9 1 1 1 3 4 8 2 2 4 1 6 1 1 2 83 6 12 1 1 216 3 1 21 1 2 2 1 5 3 4 2
2 1 1 4 1 1 2 1 6 1 5 1 6 19 11 1 1 5 2 2 2 1 2 2 7 5 2 2 377 95 4 2 1 2 1 4 3 1
2 20 1 3 3 7 1 1 1 1 1 1 1 3 2 1 1 1 1 1 5 1 400 1 1 1 1 1 6 7 1 3 1 2 3 1 2 2 1
10 1 24 28 5 3 1 3 1 7 2 78 12 2 2 5 1 6 1 4 1 24 1 1 1 1 2 50 1 1 136 1 2 1 3 2
68 1 1 2 1361 2 14 2 1 3 99 7 1 72 2 3 1 16 1 38 1 4 5 10 4 4 1 1 6 1 1 1 1 3 1
1 2 294 7 3 62 2 1 5 1 1 5 1 1 1 2 1 1 3 4 1 2 1 9 1 1 2 25 1 97 1 2 19 1 5 11 1
1 2 2 1 2 1 1 7 11 1 1 8 4 1 7 1 13 1 4 1 1 1 1 1 4 1 3 1 2 3 1 3 4 1 1 1 2 9 2
2 9 12 5 111 1 10 1 1 1 3 294 13 2 13 5 1 17 4 2 2 1 1 2 1 2 1 6 2 2 1 1 7 1 4 1
1 1 6 4 3 17 11 4 1 11 1 2 7 22 1 1 27 1 4 1 7 3 1 2 4 1 1 1 1 3 5 1 2 3 3 2 1 2
2 3 5 2 1 140 1 6 1 2 2 1 5 1 19 1 6 2 2 1 2 20 1 2 11 2 14 2 2 3 3 1 3 3 1 6 3
9 1 14 1 1 14 1 11 1 3 7 1122 3 699 3 1 1 3 1 1 1 11 1 2 18 3 4 1 3 3 7 1 1 1 1
6 1 6 9 20 2 1 1 1 5 3 2 1 1 1 18 1 8 59 1 10 1 1 1 1 2 1 4 2 9 1 5 1 325 1 54 6
1 6 3 1 821 2 23 1 10 1 4 5 5 1 8 1 17 2 1 3 2 17 1 7 1 3 2 1 1 1 2 10 1 10 3 2
1 4 4 8 34 1 2 4 2 23 1 2 2 1 12 6 14 1 1 2 1 1 1 2 3 1 12 1 1 11 4 4 4 1 2 4 2
2 1 3 4 2 1 2 1 1 2 3 7 2 2 13 1 1 2 1 26 3 1 8 1 1 4 7 1 1 3 2 1 13]
  0.959116 seconds (14.74 k allocations: 460.156 KiB)
```

```
[11]: check(z) - sqrt(2)
```

```
[11]: 0.0
```

For e, first 1000 digits:

```
[12]: @time begin
          z = cfexp(MathConstants.e, 1000)
          println(z)
      end
```

```
[2 0 2 1 2 1 1 4 1 1 6 1 1 8 1 1 10 1 1 12 1 1 11 3 2 1 3 1 73 6 1 1 1 1 1 2 31
1 1 1 2 1 1 2 1 2 15 9 1 3 1 4 2 1 2 1 2 5 5659 1 11 1 1 2 1 1 198 15 5 2 1 1 1
1 2 1 1 3 1 51 1 10 4 1 1 6 1 1 1 2 12 1 2 3 2 1 6 5 5 3 1 1 1 1 1 2 3 3 1 7 1 7
1 3 10 2 2 23 1 1 7 10 11 13 1 1 1 1 1 1 1 11 2 1 1 43 3 1 1 14 2 2 5 23 1 7 2 1
1929 1 7 12 21 1 4 3 1 3 18 1 1 3 2 11 1 2 1 4 3 1 3 4 1 1 1 1 1 2 2 1 5 1 29 1
1 30 2 1 43 1 1 4 2 1 16 2 1 2 3 4 4 1 2 1 6 3 3 5 54 11 13 2 1 17 2 33 1 1 2 19
1 7 2 3 1 20 10 9 4 1 1 2 3 23 1 2 1 71 4 1 3 1 1 61 1 1 14 4 2 1 5 2 351 2 5 2
9 2 1 2 1 3 1 2 1 5 4 4 2 1 1 1 2 1 2 3 1 2 2 9 13 1 7 1 1 13 10 10 2 1 1 1 5 2
5 1 2 9 1 14 15 1 21 1 14 1 1 34 1 12 1 1 1 1 1 2 2 1 1 2 2 1 1 1 1 2 1 1 19 2 1
1 2 14 2 4 2 2 4 1 2 3 2 1 6 2 1 394 2 1 17 2 1 20 1 1 1 3 1 3 1 1 20 1 8 1 1 3
7 1 5 23 1 3 1 39 2 1 6 1 1 1 1 1 1 1 1 2 1 2 1 2 1 2 4 1 1 1 1 1 7 3 38 1 2 1 1
1 1 8 19 1 4 1 3 1 2 10 30 2 4 2 31 29 1 261 4 1 6 1 1 34 89 1 3 2 2 3 13 15 1
52 2 7 1 5 3 1 2 5 3 1 2 1 1 1 1 4 2 2 1 4 1 5 2 1 4 2 1 3 2 15 1 4 9 2 1 1 1 3
3 23 3 4 3 2 1 11 1 6 2 6 1 1 1 6 1 1 1 7 266 3 2 1 1 2 24 2 2 1 13 1 2 2 1 1 1
1 15 2 1 1 2 2 7 2 6 2 1 5 6 1 2 1 15 4 2 1 5 1 2 1 1 8 2 3 2 1 1 2 2 13 1 312 8
2 7 1 2 4 1 2 2 3 2 1 3 1 5 2 2 25 3 269 3 5 3 4 10 1 3 6 1 1 6 6 2 1 3 5 1 3 1
1 5 1 2 4 1 17 1 1 1 1 5 1 1 8 1 1 3 3 1 2 3 18 23 2 1 20 3 3 2 1 4 8 1 3 1 1 5
1 7 3 51 8 2 3 4 5 1 4 1 4 3 55 9 7 4 1 4 1 1 5 2 2 1 2 1 3 18 1 1 1 5 1 1 1 2 1
1 5 2 6 2 39 2 2 4 1 1 3 1 40 6 3 3 2 5 2 4 10 3 1 1 1 1 2 1 2 3 11 3 5 1 1 5 2
1 1 3 1 1 16 1 1 1 3 2 3 1 1 2 1 2 1 7 10 1 21 1 1 1 1 10 13 5 1 12 1 1 11 2 8
12 2 11 1 1 1 3 1 1 1 12 1 1 7 1 1 1 6 3 23 1 2 1 2 1 13 1 1 3 1 1 4 4 1 32 1 37
2 41 1 1 4 6 2 3 1 2 1 4 11 1 13 4 1 1 3 1 8 1 4 329 1 9 1 3 2 7 1 8 17 1 5 6 1
22 1 1 5 37 1 5 4 28 1 1 2 1 3 8 3 6 4 1 3 1 5 1 3 2 1 1 1 79 1 1 1 2 5 1 1 14
10 9 14 5 1 5 2 1 2 4 4 2 1 1 2 3 8 12 12 1 57 2 1 10 4 1 4 1 1 1 5 1 17 1 1 1 7
1 1 8 1 5 1 1 7 3 7 2 1 9 18 3 1 1 3 6 1 153 1 1 8 3 2 3 37 18 2 1 1 35 7 5 1 10
2 6 1 11 1 2 4 6 3 2 1 21 5 2 2 3 1 1 4 7 1 1 1 1 37 1 18 4 1 1 2 16 11 12 1 6 1
19 1 30 82 60 1 13 22 1 7 1 1 1 3 5 10 4 1 6 2 10 2 11 2 1 2 19 11 1 1 1 2 3 3
19]
  1.060084 seconds (14.87 k allocations: 463.625 KiB)
```

```
[13]: check(z) - MathConstants.e
```

```
[13]: 0.0
```

For $\varphi = \frac{1+\sqrt{5}}{2}$:

```
[14]: @time begin
          z = cfexp(MathConstants.golden, 1000)
          println(z)
      end
```

```
[1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
2 2 1 8 2 2 2 3 2 1 2 3 2 1 1 1 15 1 4 2 1 1 2 2 1 1 2 5 1 1 6 3 2 1 8 1 54 1 1
37 5 490 1 1 1 5 1 3 6 2 1 17 1 1 1 5 1 107 1 1 7 1 1 1 3 2 1 8 1 6 2 11 7 1 2
10 11 7 3 2 37 2 4 2 1 1 3 4 1 3 3 5 4 2 1 59 1 6 5 1 1 3 1 9 15 3 1 3 1 5 1 14
7 1 1 4 2 2 20 2 3 1 49 1 8 13 2 1 115 6 1 8 1 159 18 1 4 1 1 1 3 4 1 2 1 1 3 2
3 2 2 1 1 7 6 5 1 1 13 2 2 4 4 23 5 2 3 1 3 2 3 35 9 11 1 1 5 1 2 1 5 1 4 2 3 2
2 1 3 3 4 1 54 33 13 5 1 12 13 5 2 1 2 1 1 8 2 4 1 1 3 3 1 1 1 1 6 2 1 2 2 1 1 1
1 1 2 3 6 2 1 2 1 2 1 32 3 1 1 8 2 4 43 8 1 1 1 12 1 2 2 1 1 53 2 1 4 1 1 1 1 3
1 1 1 1 1 7 2 2 4 1 73 1 9 4 4 4 7 3 1 2 1 8 1 3 2 1 1 2 1 1 1 6 1 16 1 1 3 10 1
2 6 4 1 1 2 1 17 12 1 89 3 1 1 2 9 1 4 1 11 1 2 38 1 1 1 1 1 21 1 9 3 6 1 1 11 4
3 2 2 6 1 3 2 1 2 1 1 2 12 1 2 60 7 1 2 1 1 1 1 8 1 8 4 15 3 2 1 2 1 2 3 2 1 13
1 1 3 9 1 7 15 1 1 21 1 2 3 2 1904 1 1 1 1 1 4 1 5 1 1 1 24 6 1 11 35 1 1 4 6 1
1 36 1 12 2 1 1 3 14 1 1 1 4 3 1 1 21 1 1 4 18 2 1 13 1 28 1 4 1 2 1 1 15 1 5 1
1 1 5 2 1 1 2 4 3 1 1 12 4 1 2 3 9 1 20 3 6 1 1 1 115 4 100 11 3 1 31 2 1 5 2 1
6 4 2 31 2 1 1 1 2 2 2 2 1 1 5 1 18 1 10 8 3 3 6 1 4 2 5 1 3 2 6 1 1 1 1 8 1 9 1
1 1 3 1 6 1 3 2 1 1 18 1 7 1 1 8 1 50 1 10 3 2 1 43 1 3 22 1 2 2 4 2 3 1 6 1 6 1
1 1 2 1 13 1 2 2 1 5 1 2 2 2 6 2 5 1 5 1 6 1 1 25 1041 2 1 17 1 5 2 2 1 1 3 1 26
1 5 2 1 3 1 10 3 3 19 5 2 14 2 1 1 1 2 2 1 2 5 1 2 2 1 1 100 1 2 3 1 7 3 1 3 1
17 3 1 3 6 12 1 1 1 4 3 1 5 2 4 2 13 1 3 1 1 10 2 8 1 8 68 3 1 2 1 5 1 3 1 1 22
1 1 5 5 2 4 3 8 1 6 1 5 11 1 1 2 1 1 1 9 1 1 1 31 1 6 1 1 2 1 1 1 1 1 1 2 12 1
34 20 1 131 8 4 1 4 1 9 9 2 1 17 2 3 8 7 2 12 5 2 2 1 1 1 208 3 2 9 1 2 1 1 4 3
1 13 114 1 2 1 20 26 2 1 4 1 16 4 1 1 1 4 1 2 3 2 6 4 3 1 1 2 33 4 1 31 2 1 5 1
3 10 5 1 8 1 5 1 1 2 768 1 1 2 3 1 4 1 1 1 10 1 3 3 1 8 1 4 1 4 1 1 1 3 4 1 2 1
1 5 30 1 13 1 12 1 57 1 173 1 52 2 1 2 9 2 3 2 8 8 2 2 11 6 1 5 7 1 2 2 1 2 3 1
1 1 1 1 3 4 1 16 3 2 10 1 2 10 1 1 1 3 1 7 2 3 3 2 7 34 13 2 1 1 2 33 25 3 3 7 1
53 1 49 7 6 12 3 1 1 5 2 9 1 1 9 1 22 2 1 1 2 2 4 1 3 1 1 1 1 6 1 1 1 3 3 1 1 2
1 4 2 2 2 1 1 1 2 61 1 5 1 1 31 3 5 1 7 3 4 1 41 3 3 78 6 4 1 2]
  0.789989 seconds (14.80 k allocations: 461.891 KiB)
```

```
[15]: check(z) - MathConstants.golden
```

```
[15]: 0.0
```