Zeta Function Estimations

Colton Williams

September 20, 2017

Values

The following tables contain the results of the zeta_forward and zeta_backward functions.

Results for $s = 2$

Expression	$N = 10^4$	$N = 10^5$	$N = 10^6$
$forward$	1.644725	1.644725	1.644725
$backward$	1.644834	1.644924	1.644933
$\|forward - backward\|$	0.000109	0.000199	0.000208
$\|zeta(2) - forward\|$	0.000209	0.000209	0.000209
$\|zeta(2) - backward\|$	0.000100	0.000010	0.000001
$ratio$	0.478469	0.047847	0.004785

Results for $s = 3$

Expression	$N = 10^4$	$N = 10^5$	$N = 10^6$
$forward$	1.202051	1.202051	1.202051
$backward$	1.202057	1.202057	1.202057
$\|forward - backward\|$	0.000006	0.000006	0.000006
$\|zeta(3) - forward\|$	0.000006	0.000006	0.000006
$\|zeta(3) - backward\|$	0.000000	0.000000	0.0000010
$ratio$	$undef$	$undef$	$undef$

Conclusion

The zeta function, when evaluated backwards (from the smallest term first), was much more accurate than when evaluated forwards. The backwards sum was always closer to the actual value, and when $s = 3$, it was equivalent to the actual value of the zeta function to within six decimal digits of precision. Both the forward and backward functions were much more accurate just from stepping from $s = 2$ to $s = 3$.

Code

```c
/*
        COLTON WILLIAMS 2017
        NUMERICAL ANALYSIS
        ZETA FUNCTION
*/

#define ZETA2 1.6449340668482264
#define ZETA3 1.20205690315959

#include <stdio.h>
#include <math.h>

float zeta_forward(int s, int n)
{
        float total = 0.0;
        int N = (int)pow(10, n);
        for (int i = 1; i < N + 1; ++i)
        {
                total = total + 1.0/(pow(i, s));
        }
        return total;
}

float zeta_backward(int s, int n)
{
        float total = 0.0;
        int N = (int)pow(10, n);
        for (int i = N + 1; i > 0; --i)
        {
                total = total + 1.0/(pow(i, s));
        }
        return total;
}

int main() {
        printf("ENTER A VALUE FOR S :: ");
        int s, n;       scanf("%d", &s);
        printf("ENTER A VALUE FOR N :: ");
        scanf("%d", &n);
        printf("zeta_forward(%d) \t%f\n", s, zeta_forward(s, n));
        printf("zeta_backward(%d)\t%f\n", s, zeta_backward(s, n));
        printf("zeta(2)-forward  \t%f\n", ZETA2 - zeta_forward(s, n));
        printf("zeta(2)-backward \t%f\n", ZETA2 - zeta_backward(s, n));
        printf("zeta(3)-forward  \t%f\n", ZETA3 - zeta_forward(s, n));
        printf("zeta(3)-backward \t%f\n", ZETA3 - zeta_backward(s, n));
        return 0;
}
```