cMTS Profile

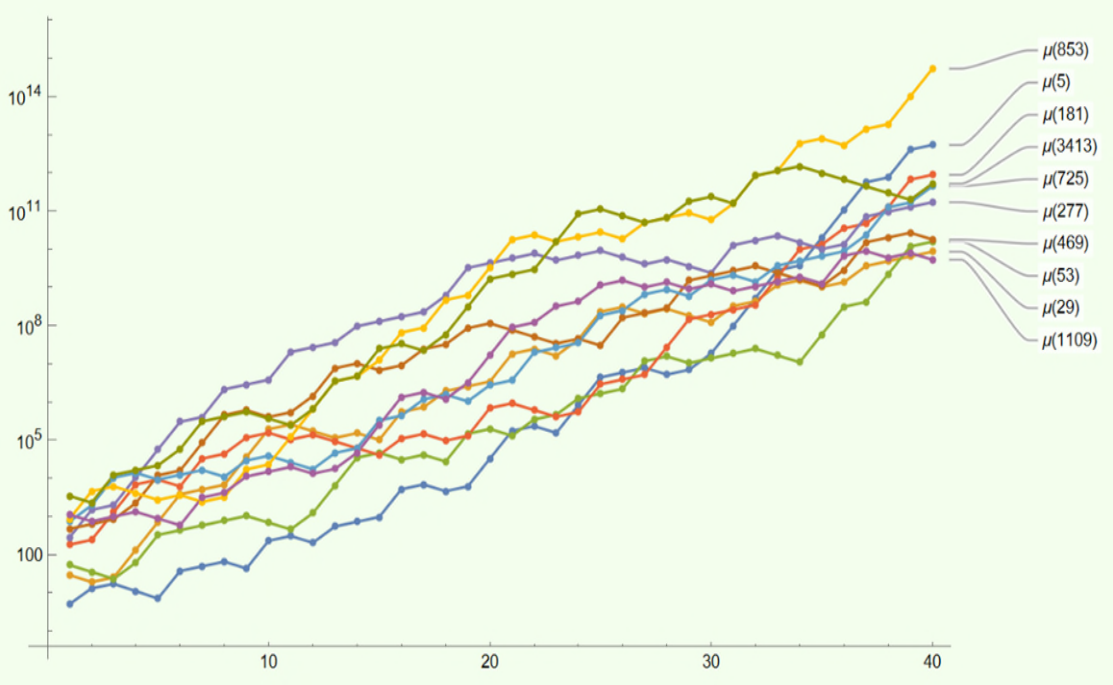
Background and purpose

This file has been converted from a Mathematica notebook, and partially edited for clarity.

The purpose of this notebook is to document key statistical elements of closed Minimum track sequences (cMTS). These are sequences designed to mirror Collatz sequences in reverse. I have demonstrated in my paper, *New Perspectives on the Collatz Sequence.pdf* that Collatz sequences are composed of one or more segments of cMTS. It is thus important to document the statistical behavior of cMTS since that has a bearing on the probabilistic trajectory of the Collatz sequences that track them. Background information and summaries of critical concepts used in this notebook can be found in the file, *Collatz Sequences and the RCS.pdf.*. This files, as well as *New Perspectives,* can be downloaded from my website, BandInOne.com.

Like Collatz sequences, cMTS can appear to move arbitrarily for a time, but inevitably and necessarily they must rise, and rise significantly. Below is simple graphic of ten cMTS. Note: the vertical scale is logarithmic so the ascent is even more pronounced than the slope indicates. The cMTS are *not* composite, and they have no upper bound. It is the purpose of this paper to document certain statistical aspects of cMTS. I also wanted to verify that there is no correlation between (a) the position of a neighborhood and its cMTS, or the HN level of cBases, and (b) the statistical performance of those cMTS. While there is no reason to suspect a different statistical performance based on these factors, it is useful to verify that fact.

Graph of ten cMTS



Organization of this file

The following section, **Methods and Findings**, include notes on the methods and data structure, as well as a summary of the findings. The **Data** section that follows contains the Mathematica coding and output. This is followed by the **Functions** section which contains notes on the coding.

Methods and Findings

Method and Data structure

The focus of this report is the statistical profile of cMTS. The function **mtsProfile** begins with a random sample of 10 K HN. For each of these HN it calculates their cMTS, and from each of those sequences randomly selects a term position (index between 1 and 20) to build a neighborhood. It then identifies the first five HN^C (non-S_0) in that neighborhood and computes their cMTS. For each of these sequences it records the value of the term in the 10^{th}, and 20^{th} positions, and compares them with the cBase (the first term of that cMTS).

- If the cBase is lower than the distal term, it computes the quotient (distal term/cBase) , the factor by which the distal term exceeds the cBase.
- If the cBase is higher than the distal term, it computes the quotient (cBase/distal term), the factor by which the cBase exceeds the distal term*; these cases are marked as negative*.

Given that the cMTS of five HN are evaluated for each of the original 10 K HN, the total sample size is 50 K per run. The data was generated in five separate runs. Each run had a characteristic sample range, form 10^6 to 10^30.

Here is a sample of the data generated by mtsProfile.

P-Index	cBase	N-Indec	Q10±	Q20±
3	3077083	1	426.185	7568.08
3	12308333	2	426.185	60544.7
3	196933333	3	26.6366	59.1257
3	787733333	4	-2.40271	118.251
3	12603733333	5	6818.97	6.19977*10^7
4	23487517	1	26.6366	60544.7
4	93950069	2	53.2732	1892.02
4	1503201109	3	106.546	121089.
4	6012804437	4	106.546	15136.2
4	96204870997	5	1704.74	1.93743*10^6

The position (P-index) repeats because the function calculates the cMTS for *five* neighbors of the base at each randomly chosen position. The N-index can be misleading. If you have read my paper above, you would know that, technically, an N_1 cannot be an HN or form a cMTS. The indexing in this report is not "technical" indexing but *relative* indexing: the code finds the first five neighbors of a base that *will* form a cMTS (thus excluding S_0 bases). I am not interested in documenting the specific HN_i in this study. The focus here is the statistical profile of cMTS and its relation to term position or neighborhood. A separate study, the Collatz Profile, identifies specific HN_i and documents their frequency.

The function **rptByIndex** analyzes the profile data by the *relative* index of the HN. Here is an example of its output.

Index	Count	T10+ Per	T10+ Avg	T10- Avg	T20+ Per	T20+ Avg	T20-Avg
1	10000	96.7%	3380.49	-2.7522	99.6%	8.23816*10^6	-2.46696
2	10000	95.9%	2024.33	-2.81485	99.5%	9.31469*10^6	-4.28813
3	10000	96.5%	2584.1	-2.54466	99.7%	1.72222*10^7	-1.85561
4	10000	95.7%	1794.4	-2.64074	99.6%	1.45873*10^7	-3.90735
5	10000	96.6%	3203.67	-2.35993	99.8%	1.1039*10^7	-2.85764
Mean:		96.3%	2597.4	-2.622	99.6%	1.20799*10^7	-3.075
SD:			698.959	0.18		3.74912*10^6	1.009

The function **rptByPos** analyzes the data by the term position. Below is an example of this output. Note: the counts by position vary because the position was randomly chosen (between 1 and 20). However, the total of all positions will be 50 K, thus the constant average of 2500 (an integrity check).

Pos	Count	T10+ Per	T10+ Avg Q	T10- Avg Q	T20+ Per	T20+ Avg Q	T20-Avg
1	2475	96.6%	2792.96	-2.41701	99.6%	7.37917*10^6	-1.86967
2	2410	96.1%	3195.67	-2.58356	99.6%	3.45218*10^7	-3.13908
3	2655	95.9%	2382.2	-2.87664	99.5%	1.25477*10^7	-2.16488
4	2390	96.1%	1914.36	-2.79024	99.6%	1.25496*10^7	-3.84868
5	2625	96%	2323.66	-2.44892	99.7%	2.23212*10^7	-5.81812
6	2525	96.2%	3478.74	-2.64048	99.6%	1.37844*10^7	-2.28515
7	2470	95.9%	2276.26	-2.6525	99.8%	8.91661*10^6	-2.34529
8	2385	96.3%	2286.84	-2.41621	99.7%	9.52033*10^6	-1.39171
9	2460	96.7%	2309.45	-2.1976	99.7%	8.79427*10^6	-3.09269
10	2550	96.6%	2533.17	-2.44414	99.6%	8.32827*10^6	-2.46009
11	2455	96.4%	2117.49	-2.98974	99.6%	5.53775*10^6	-5.05139
12	2655	96.5%	1997.08	-2.23657	99.7%	4.12974*10^6	-4.46507
13	2490	96.3%	2050.81	-2.81196	99.6%	5.97474*10^6	-3.89679
14	2330	96.7%	2079.	-2.24669	99.7%	8.45209*10^6	-1.98447
15	2380	96.1%	2457.72	-2.57064	99.6%	9.31161*10^6	-1.92434
16	2725	96.1%	2129.16	-2.56289	99.4%	1.25264*10^7	-3.24732
17	2395	95.7%	5679.43	-2.96805	99.5%	1.57827*10^7	-6.74444
18	2515	96.3%	2645.54	-3.13397	99.6%	1.11481*10^7	-3.44413
19	2470	96.4%	2843.81	-2.82591	99.6%	1.87932*10^7	-1.6838
20	2640	96.7%	2667.5	-2.62622	99.7%	1.16215*10^7	-2.43549
Mean:	2500.	96.3%	2608.04	-2.622	99.6%	1.20971*10^7	-3.165
SD:	110.299		827.453	0.263		6.88073*10^6	1.444

A curious Pattern

I have further documented in all the run the ten maximum factors for the $T10\pm$ and $T20\pm$ categories. Here is the output from the first run for the $T10\pm$.

Base	T10+ High Factors	Base	T10- High Factors
427579564406101	3.49131*10^6	1732143365	-38.4434
23873965	3.49131*10^6	68607557	-19.2217
23236703573	1.74566*10^6	161515781	-19.2217
1959865032037717	872828.	214765829	-19.2217
466913387894101	872828.	433035841	-19.2217
597327478405	872828.	701932229	-19.2217
285766619947	872828.	856667825	-19.2217
233260839637	872828.	1671906005	-19.2217
196194047317	872828.	2603071829	-19.2217
124107277435	872828.	4231879445	-19.2217

There is a false pattern and a real pattern in this data The false pattern: the repeated high factors are not truly identical since they are rounded off. For example.

- The 10th term of μ(427 579 564 406 101) is 1 492 813 287 049 436 876 485. Dividing the first by the second you get: 3.4913111180206263089×10^6
- The 10th term of μ(23 873 965) is 83 351 439 167 173. Dividing the first by the second you get: 3.4913111067714558516×10^6

The real pattern: although the unrounded numbers are different, they are obviously very close to the rounded numbers. There appears to be an approximate multiplicative pattern of 2 or 4. Throughout the runs you will see the same high quotients, rounded numbers that generally fit a multiplicative factor of 2 or 4. Although I was initially surprised by this regularity, (it prompted me to verify the logic of the coding), I

have concluded that it is inherent "in the system". Although I have not investigated this pattern further, I believe it is based on the dominance of 4 and 2 in the various formulas, i.e. neighborhoods $(4n+1)$, the R function, $(4n-1)/3$ and $(2n-1)/3$, etc.

Findings

I have summarized the data from all runs in the tables below. The data in both reports show a high degree of consistency, both within runs (by position or index), and in the combined runs. There is no indication that neighborhood term position or neighbor index affects cMTS statistical performance. All cMTS, no matter where they originate, tend toward the same average patterns and values.

Report by Position:

Run	Count Span	T 10+ Per Mean Q Span SD	T 10 - Per Mean Q Span SD	T 20+ Per Mean Q Span SD	T 20 - Per Mean Q Span SD
R 1: 10^6	2500 2330 | 2725	96.3% 2608 1914 | 5679 827	3.7% -2.63 –2.19 | – 3.13 0.26	99.6% 1.2×10^{7} 4.1×10^{6} | 3.45×10^{7} 6.8×10^{6}	0.4% –3.16 –1.39 | –5.81 1.144
R 2: 10^12	2500 2275 |2715	96.5% 2501 1551-5509 728	3.5% –2.67 –2.19 –3.66 0.38	99.7 4.7×10^{7} 4.2×10^{6} | 21.7×10^{7} 4.65×10^{7}	0.3% –2.634 –1.23 | –4.69 1.08
R 3: 10^18	2500 2320 | 2655	96.5% 2775 1970 | 4390 633	3.5% –2.87 –2.63 | –3.95 0.37	99.6% 1.77×10^{7} 4.4×10^{6} | 6.4×10^{7} 1.54×10^{7}	0.4% –5.58 –1.66 |–37.6 8.26
R 4: 10^24	2500 2280 | 2730	96.2% 2813 2122 | 4734 657	3.8% –2.54 –2.22 |–-2.87 0.19	99.6% 1.62×10^{7} 5.2×10^{6} | 8.3×10^{7} 2.6×10^{7}	0.4% –3.01 -1.51 | -6.9 1.53
R 5: 10^30	2500 2310 | 2710	96.3% 2531 2117 | 3762 455	3.7% –2.55 –2.12 | –2.87 0.22	99.7% 1.34×10^{7} 4.2×10^{6} | 2.8×10^{7} 7.3×10^{6}	0.3% –3.13 –1.35| –6.8 1.43
Mean Mean Span SD	2500	96.36% 2645 1551 | 5679 660	3.64% –2.65 –2.12 | –3.95 0.28	99.65% 2.23×10^{7} 4.1×10^{6} | 21.7×10^{7} 2.04×10^{7}	0.35% –3.5 –1.23 | –37.6 2.68

The count for each position was determined randomly. The mean of 2500 reflects the total of 50 K in the sample. The span of min or max are averages of the term data. I did not do SD for this item.

In looking at the summaries of individual runs, we can see that no term position has unique characteristics. The highs and lows move around, i.e. they are not related to a particular position or index. (Please refer to the **rptByPos** and **rptByIndex** outputs under each run for this data.)

The $T10\pm$ and $T20\pm$ percentages are remarkably consistent from run to run, although the average quotients in each category (and in each run) are highly variable. However, even with this variability, the quotients for the positive values are orders of magnitude greater than the quotients for negative values. Over 96% of the cMTS are an average of 2645 times greater than their starting cBase after 10 terms, and 99% are 20 million times greater than their starting cBase at the 20th position. For the negative terms, roughly 3.5% are less than their cBase at the 10th term by an average factor of -2.65, and at the 20th term

0.35% are less than their cBase by an average factor of −3.65. These figures underscore the strong tendency of cMTS to rise, and a tracking Collatz to fall in value.

Report by Index

Run	Count Span	T 10+ Per Mean Q Span SD	T 10 - Per Mean Q Span SD	T 20+ Per Mean Q Span SD	T 20 - Per Mean Q Span Averages SD
R 1: 10^6	10 000	96.7% 2597 1794 | 3380 698	3.3% −2.62 −2.365 | −2.81 0.18	99.6% 1.2 ×10^7 8.3 ×10^6 | 1.72×10^7 3.7 ×10^6	0.4% −3.08 −1.85 | −4.28 1.09
R 2: 10^12	10 000	96.9% 2497 2056 | 2893 302	3.1/% -2.8 −2.41 | −2.84 0.18	99.6% 2.79×10^7 7.5 ×10^6 | 7.02×10^7 2.67×10^7	0.4% −2.62 −2.07 | −3.2 0.43
R 3: 10^18	10 000	96.3% 2780 2223 | 3345 402	3.7% −2.87 −2.67 | −2.75 0.16	99.6% 1.76×10^7 9.10 ×10^6 | 2.44×10^7 6.1 ×10^6	0.4% −7.85 −1.96 | −11.01 3.67
R 4: 10^24	10 000	96.2% 2811 1999 | 3338 504	3.8% −2.54 −2.31 | −3.04 0.158	99.6% 1.57×10^7 8.3 ×10^6 | 3.6×10^7 1.014 ×10^7	0.4% −3.07 −2.16 | −3.97 1.088
R 5: 10^30	10 000	96.3% 2524 2225 | 2741 248	3.7% −2.53 −2.42 | −2.84 0.18	99.7% 1.33 ×10^7 9.6 ×10^6 | 2.04×10^7 4.5 ×10^6	0.3% −3.24 −2.77 | −5.85 4.79
Mean Mean Avg Span SD	10 000	96.58% 2641 1794 | 3380 430	3.42% −2.67 −2.43| −2.86 .17	99.62% 1.73×10^7 8.56 ×10^6 | 3.36×10^7 5.8 ×10^6	0.38% −3.97 −2.16 | −5.66 2.21

The report by Index shows the same characteristics and patterns. This report looks at the same data but is organized by HN index; it was designed to determine if index affects other parameters, and the data shows that it does not.

High Quotients

	T 10 + High Quotients	T 10 − High Quotients	T 20+ High Quotients	T 20 − High Quotients
Run 1: 10^6	3.49 × 10^6(2) 1.74 × 10^6 (1) 8.73 × 10^5 (7)	−38.44 (1) −19.22 (9)	6. 34 × 10^{10} (1) 3.17 × 10^{10} (1) 1.58 ×10^{10} (4) 7.93 ×10^9 (4)	−34.64 (1) −17.32 (9)
Run 2: 10^12	3.49 × 10^6 (2) 8.73 × 10^5 (8)	−38.44 (1) −19.22 (9)	5.07 × 10^{11} (2) 6. 34 × 10^{10}(2) 3. 17 × 10^{10} (2) 1. 58 ×10^{10} (4)	−17.32 (4) −8.56 (6)
Run 3: 10^18	3.49 × 10^6 (3) 1.74 × 10^6 (1) 8.73 × 10^5 (6)	−38.44 (1) −19.22 (9)	6. 34 × 10^{10} (3) 3. 17 × 10^{10} (6) 1. 58 ×10^{10} (1)	−277.105 (1) −69.276 (1) −34.64 (5) −17.32 (1) −8.56 (2)
Run 4: 10^24	3.49 × 10^6 (2) 1.74 × 10^6 (1) 8.73 × 10^5 (7)	−38.44 (1) −19.22 (9)9)	2.53 × 10^{11} (1) 1. 58 ×10^{10} (6) 7.931 ×10^9 (3)	−34.64 (2) −17.32 (5) −8.56 (3)
Run 5: 10^30	8.73 × 10^5 (10)	−19.2 (10)	3. 17 × 10^{10} (5) 1. 58 ×10^{10} (5)	−34.64(1) −17.3 (5) −8.56 (5)

This is a log of the top 10 values in each category with a parenthetical count. These figures are taken from output which screened for the top 10 figures so the counts don't go beyond 10. The highest negative value for $T20$ (-277) is still so much smaller than the *average* factor for the $T20$ positives, 1.73×10^7.

See *A curious pattern* above for a discussion of the rough pattern of multiples of 2 or 4.

Data

Run 1: 10^6, Sample Size: 50K

```
r1=randSamp[10^6, 10^7, 10000];
d1= mtsProfile[r1];
```

This is a sample of the data structure. The Position index repeats because 5 neighbors are selected.

```
TableForm[Take[d1,10], TableHeadings→{None,{"P-Index","cBase", "N-Index", "Q10±", "Q20±"}}]
```

P-Index	cBase	N-Index	Q10±	Q20±
3	3077083	1	426.185	7568.08
3	12308333	2	426.185	60544.7
3	196933333	3	26.6366	59.1257
3	787733333	4	-2.40271	118.251
3	12603733333	5	6818.97	6.19977 X 10^7
4	23487517	1	26.6366	60544.7
4	93950069	2	53.2732	1892.02
4	1503201109	3	106.546	121089.
4	6012804437	4	106.546	15136.2
4	96204870997	5	1704.74	1.93743 X 10^6}

Report by Position

```
rptByPos[d1]
```

Pos	Count	T10+ Per	T10+ Avg Q	T10- Avg Q	T20+ Per	T20+ Avg Q	T20-Avg
1	2475	96.6%	2792.96	-2.41701	99.6%	7.37917*10^6	-1.86967
2	2410	96.1%	3195.67	-2.58356	99.6%	3.45218*10^7	-3.13908
3	2655	95.9%	2382.2	-2.87664	99.5%	1.25477*10^7	-2.16488
4	2390	96.1%	1914.36	-2.79024	99.6%	1.25496*10^7	-3.84868
5	2625	96%	2323.66	-2.44892	99.7%	2.23212*10^7	-5.81812
6	2525	96.2%	3478.74	-2.64048	99.6%	1.37844*10^7	-2.28515
7	2470	95.9%	2276.26	-2.6525	99.8%	8.91661*10^6	-2.34529
8	2385	96.3%	2286.84	-2.41621	99.7%	9.52033*10^6	-1.39171
9	2460	96.7%	2309.45	-2.1976	99.7%	8.79427*10^6	-3.09269
10	2550	96.6%	2533.17	-2.44414	99.6%	8.32827*10^6	-2.46009
11	2455	96.4%	2117.49	-2.98974	99.6%	5.53775*10^6	-5.05139
12	2655	96.5%	1997.08	-2.23657	99.7%	4.12974*10^6	-4.46507
13	2490	96.3%	2050.81	-2.81196	99.6%	5.97474*10^6	-3.89679
14	2330	96.7%	2079.	-2.24669	99.7%	8.45209*10^6	-1.98447
15	2380	96.1%	2457.72	-2.57064	99.6%	9.31161*10^6	-1.92434
16	2725	96.1%	2129.16	-2.56289	99.4%	1.25264*10^7	-3.24732
17	2395	95.7%	5679.43	-2.96805	99.5%	1.57827*10^7	-6.74444
18	2515	96.3%	2645.54	-3.13397	99.6%	1.11481*10^7	-3.44413
19	2470	96.4%	2843.81	-2.82591	99.6%	1.87932*10^7	-1.6838
20	2640	96.7%	2667.5	-2.62622	99.7%	1.16215*10^7	-2.43549
Mean:	2500.	96.3%	2608.04	-2.622	99.6%	1.20971*10^7	-3.165
SD:	110.299		827.453	0.263		6.88073*10^6	1.444

Report by HN Index

```
rptByIndex[d1]
```

Index	Count	T10+ Per	T10+ Avg	T10- Avg	T20+ Per	T20+ Avg	T20-Avg
1	10000	96.7%	3380.49	-2.7522	99.6%	8.23816*10^6	-2.46696
2	10000	95.9%	2024.33	-2.81485	99.5%	9.31469*10^6	-4.28813
3	10000	96.5%	2584.1	-2.54466	99.7%	1.72222*10^7	-1.85561
4	10000	95.7%	1794.4	-2.64074	99.6%	1.45873*10^7	-3.90735
5	10000	96.6%	3203.67	-2.35993	99.8%	1.10369*10^7	-2.85764
Mean:		96.3%	2597.4	-2.622	99.6%	1.20799*10^7	-3.075
SD:			698.959	0.18		3.74912*10^6	1.009

Report of maximum factors for Pos 10

T105HighFac[d1]

Base	T10+ High Factors	Base	T10- High Factors
427579564406101	3.49131*10^6 1	732143365	-38.4434
23873965	3.49131*10^6	68607557	-19.2217
23236703573	1.74566*10^6	161515781	-19.2217
1959865032037717	872828.	214765829	-19.2217
466913387894101	872828.	433035841	-19.2217
597327478405	872828.	701932229	-19.2217
285766619947	872828.	856667825	-19.2217
233260839637	872828.	1671906005	-19.2217
196194047317	872828.	2603071829	-19.2217
124107277435	872828.	4231879445	-19.2217

Report of maximum factors for Pos 20

T20HighFac[d1]

Base	T20+ High Factors	Base	T20- High Factors
6296061269	6.34857*10^10	448742354197	-34.6381
566211029	3.17428*10^10	214765829	-17.3191
1959865032037717	1.58714*10^10	68607557	-17.3191
457297951109461	1.58714*10^10	1732143365	-17.319
1524332527957	1.58714*10^10	4757599921	-17.319
735105749	1.58714*10^10	27847628117	-17.319
93956875662677	7.93571*10^9	45097759061	-17.319
175729688917	7.93571*10^9	71780923925	-17.319
189797063189	7.93571*10^9	161182063957	-17.319
114825074005	7.93571*10^9	433505425985	-17.319

Run 2: 10^12, Sample Size: 50K

```
r2=randSamp[10^12, 10^13, 10000];
d2= mtsProfile[r2];
```

Report by Position

```
rptByPos[d2]
```

Pos	Count	T10+ Per	T10+ Avg Q	T10- Avg Q	T20+ Per	T20+ Avg Q	T20-Avg
1	2490	96.7%	5059.36	-2.5764	99.6%	2.17413*10^8	-4.03455
2	2560	96.1%	1992.05	-3.02741	99.6%	1.08494*10^7	-4.00503
3	2455	96%	2472.93	-2.78665	99.4%	1.55769*10^7	-1.94839
4	2520	96.5%	1995.2	-2.40271	99.7%	1.14394*10^7	-1.23707
5	2515	96.3% 2	523.24	-2.19378	99.9%	7.58606*10^6	-4.69057
6	2535	96.7%	2320.97	-2.21455	99.8%	2.68249*10^7	-4.51017
7	2500	96.7%	2070.57	-3.66267	99.8%	9.52121*10^6	-1.80407
8	2685	96%	2310.22	-2.4472	99.3%	1.24121*10^7	-2.64597
9	2455	96.8%	2940.36	-2.38731	99.7%	1.47943*10^7	-2.7061
10	2305	96.8%	2575.32	-2.69493	99.6%	7.19613*10^6	-3.03083
11	2275	96.4%	1550.35	-2.4917	99.6%	4.19797*10^6	-1.92434
12	2450	96.4%	2379.83	-3.31408	99.6%	5.66475*10^6	-2.06648
13	2650	96.8%	2189.2	-2.2883	99.8%	1.39911*10^7	-1.89427
14	2705	95.7%	2135.56	-2.65343	99.7%	5.49681*10^7	-2.47415
15	2490	96.7%	2087.04	-2.83282	99.7%	1.43383*10^7	-1.35305
16	2455	96.5%	2650.09	-3.01045	99.6%	3.84409*10^7	-1.84015
17	2715	96.5%	2393.26	-2.95333	99.6%	8.34782*10^6	-4.22152
18	2360	96.5%	2236.43	-2.46131	99.7%	1.81622*10^7	-1.89427
19	2480	96.9%	2563.89	-2.83397	99.8%	4.00453*10^7	-2.7061
20	2400	96.4%	3571.95	-2.26462	99.7%	2.54582*10^7	-1.70098
Mean:	2500.	96.5%	2500.89	-2.675	99.7%	2.78614*10^7	-2.634
SD:	120.928		728.707	0.385		4.65259*10^7	1.088

Report by Index

```
 rptByIndex[d2]
```

Index	Count	T10+ Per	T10+ Avg	T10- Avg	T20+ Per	T20+ Avg	T20-Avg
1	10000	96.9%	2607.16	-2.56443	99.6%	7.5217*10^7	-2.0721
2	10000	95.7%	2056.49	-2.82375	99.5%	1.84523*10^7	-2.86396
3	10000	96.9%	2893.83	-2.41437	99.7%	1.99007*10^7	-2.44551
4	10000	96%	2465.21	-2.84876	99.6%	8.90588*10^6	-2.53449
5	10000	96.9%	2464.26	-2.61266	99.8%	1.73432*10^7	-3.20026
Mean:		96.5%	2497.39	-2.653	99.6%	2.79638*10^7	-2.623
SD:			302.382	0.183		2.67598*10^7	0.428

Report of maximum factors for Pos 10

```
t10HighFac[d2]
```

Base	T10+ Max Factors	Base	T10- Max Factors
5648910530858374464853	3.49131*10^6	10575928881386837	-38.4434
353910009331285	3.49131*10^6	2790075803165	-19.2217
108252919472215381	872828.	4380885657233	-19.2217
8588492141696341	872828.	23061446998949	-19.2217
2364305393753580565	872828.	75477686383349	-19.2217
527193494207010133	872828.	99630029295701	-19.2217
4550992415203669	872828.	161800452414197	-19.2217
2000945652282277	872828.	522446711939783	-19.2217
4222698166063	872828.	606652115544533	-19.2217
9613818425656661	436414.	1188422672888069	-19.2217

Report of maximum factors for Pos 20

t20HighFac[d2]

Base	T20+ Max Factors	Base	T20- Max Factors
4222698166063	5.07886*10^11	69249314930861	-17.319
72371097359557	1.26971*10^11	44283279510389	-17.319
17951840776709758408021	6.34857*10^10	52430225530078549	-17.319
4140140289764117	6.34857*10^10	16660692625969393194325	-17.319
7094046826981376341	3.17428*10^10	124947606855637	-8.65952
16010680588079	3.17428*10^10	1207729836633941	-8.65952
5533219529056872184597	1.58714*10^10	8379203566511125	-8.65952
18261085655975893	1.58714*10^10	23340699019879445	-8.65952
527193494207010133	1.58714*10^10	11622654806338901	-8.65952
643955335848981205	1.58714*10^10	41028287816537429	-8.65952

Run 3: 10^18, Sample Size: 50K

```
r3=randSamp[10^18, 10^19, 10000];
d3= mtsProfile[r3];
```

Report by Position

```
rptByPos[d3]
```

Pos	Count	T10+ Per	T10+ Avg Q	T10- Avg Q	T20+ Per	T20+ Avg Q	T20-Avg
1	2545	96.5%	2726.28	-2.85656	99.5%	1.05895*10^7	-5.24567
2	2455	96.4%	2404.77	-2.93513	99.6%	4.38721*10^6	-2.28515
3	2655	95.9%	1994.34	-3.03676	99.8%	2.0805*10^7	-3.03083
4	2380	95.8%	3415.31	-2.55889	99.3%	5.06349*10^7	-2.23253
5	2465	96.1%	2831.25	-2.83619	99.7%	1.01222*10^7	-37.6148
6	2425	96.2%	3416.03	-2.73275	99.6%	2.09226*10^7	-2.04461
7	2430	96.5%	2493.44	-2.67445	99.5%	9.01736*10^6	-2.36169
8	2460	96.3%	1970.29	-2.68999	99.8%	1.32077*10^7	-3.60813
9	2865	96.5%	3017.32	-3.51912	99.5%	8.97135*10^6	-2.45353
10	2545	95.8%	3836.82	-2.75405	99.6%	7.75209*10^6	-1.6838
11	2515	96.4%	2460.61	-2.26923	99.6%	6.37186*10^7	-8.1183
12	2445	96%	2335.68	-3.95085	99.4%	1.3056*10^7	-2.01025
13	2320	96.5%	2178.59	-2.56387	99.7%	4.93573*10^6	-16.082
14	2540	96.9%	2979.55	-2.86477	99.7%	1.14403*10^7	-4.60037
15	2440	96.6%	2495.09	-2.90327	99.6%	3.55857*10^7	-3.13908
16	2600	97.4%	4390.75	-2.65374	99.7%	1.47012*10^7	-6.95855
17	2565	96.2%	2192.91	-2.89811	99.6%	1.16602*10^7	-2.05664
18	2390	96.6%	3052.03	-3.27777	99.9%	1.5938*10^7	-1.80407
19	2425	95.7%	2260.4	-2.76081	99.5%	4.96857*10^6	-2.43549
20	2535	95.7%	3059.03	-2.63206	99.5%	2.20536*10^7	-1.66529
Mean:	2500.	96.3%	2775.53	-2.868	99.6%	1.77234*10^7	-5.572
SD:	118.121		633.511	0.367		1.54776*10^7	8.261

Report by Index

```
rptByIndex[d3]
```

Index	Count	T10+ Per	T10+ Avg	T10- Avg	T20+ Per	T20+ Avg	T20-Avg
1	10000	96.5%	3345.76	-2.78605	99.6%	2.02185*10^7	-1.96548
2	10000	96.1%	2223.85	-2.79805	99.7%	1.38028*10^7	-2.60462
3	10000	96.7%	2794.38	-3.03462	99.6%	2.44139*10^7	-11.0092
4	10000	95.6%	2675.78	-3.04325	99.6%	2.07123*10^7	-5.36066
5	10000	96.6%	2863.39	-2.67639	99.6%	9.10119*10^6	-3.32464
Mean:		96.3%	2780.63	-2.868	99.6%	1.76498*10^7	-4.853
SD:			402.308	0.163		6.11573*10^6	3.67

Report of maximum factors for Pos 10

```
t10HighFac[d3]
```

Base	T10+ High Factors	Base	T10- High Factors
1534061058188819650225	3.49131*10^6	9662524786804328345834837	-38.4434
47071243168838045842199893	3.49131*10^6	53295318299041724753	-19.2217
67425275905682365629781	3.49131*10^6	74985503819430110021	-19.2217
24146922076507341113	1.74566*10^6	719738407055868783317	-19.2217
11884715844835174414921920 85	872828.	4388094727538250401621	-19.2217
70415826980906134758409 45	872828.	2415631196701082086458709	-19.2217
13069523581114275424290 13	872828.	3694238371068699167	-19.2217
568805051801277293143381	872828.	6614855003394597203	-19.2217
459309316541821240123861	872828.	10692914489282270501	-19.2217
78904885565269172409685	872828.	13620432655990390841	-19.2217

Report of maximum factors for Pos 20

`t20HighFac`[d3]

{Base,	T20+ Max Factors,	Base,	T20- Max Factors},
14779037712798895441237	6.34857*10^10	1418502635195311555157	-277.105
994107975681483211365205	6.34857*10^10	140412072250747574127445	-69.2762
21026337802265450795	6.34857*10^10	8767854916416169252181	-34.6381
2393154677336556766942933	3.17428*10^10	119758386288702750634325	-34.6381
2495035223469090174888725	3.17428*10^10	17291756676120632344128853	-34.6381
41057826002247929349461	3.17428*10^10	22716012734604500255346773	-34.6381
450289528151879335093	3.17428*10^10	31108956261740361693770069	-34.6381
5878780318171117960883285	3.17428*10^10	1405744296633743600786773	-17.319
67425275905682365629781	3.17428*10^10	3743721180470506697045	-8.65952
227810547306107250718037	1.58714*10^10	7507477083296223175	-8.65952

Run 4: 10^24, Sample Size: 50K

```
r4=randSamp[10^24, 10^25, 10000];
d4= mtsProfile[r4];
```

Reporet by Position

```
rptByPos[d4]
```

Pos	Count	T10+ Per	T10+ Avg Q	T10- Avg Q	T20+ Per	T20+ Avg Q	T20-Avg
1	2590	96.4%	2770.36	-2.61164	99.7%	1.44341*10^7	-6.90056
2	2465	95.7%	2581.74	-2.59722	99.7%	1.26883*10^7	-2.30019
3	2600	96.3%	2487.53	-2.47702	99.7%	1.28737*10^7	-5.81812
4	2410	96.1%	2473.41	-2.87061	99.6%	1.32422*10^7	-2.04461
5	2610	96.3%	4065.9	-2.62796	99.8%	7.49382*10^6	-1.44325
6	2450	96.3%	4734.59	-2.81651	99.4%	5.9972*10^6	-2.88651
7	2540	96.5%	2183.07	-2.32172	99.8%	7.86745*10^6	-2.59786
8	2405	96.3%	2929.1	-2.4828	99.6%	1.63013*10^7	-5.4122
9	2280	96.1%	2570.98	-2.36266	99.4%	1.26693*10^8	-2.7061
10	2560	96.9%	2522.82	-2.66123	99.8%	1.52003*10^7	-1.80407
11	2425	96.5%	2363.86	-2.51578	99.5%	8.1886*10^6	-2.36169
12	2730	96.2%	2259.32	-2.78028	99.9%	6.97053*10^6	-1.89427
13	2475	96.4%	3166.13	-2.26773	99.7%	8.39927*10^6	-3.09269
14	2405	96.3%	3084.71	-2.49615	99.3%	1.13237*10^7	-1.48836
15	2485	96.1%	2826.94	-2.77047	99.5%	1.56952*10^7	-2.33141
16	2610	95.8%	3102.47	-2.27045	99.7%	1.36861*10^7	-2.30019
17	2465	96.4%	2122.1	-2.22524	99.6%	8.80956*10^6	-1.56352
18	2480	95.9%	2192.95	-2.49787	99.4%	6.82407*10^6	-3.78854
19	2540	96%	3384.62	-2.55734	99.4%	5.24674*10^6	-2.78342
20	2475	95.4%	2445.91	-2.72165	99.7%	7.12128*10^6	-4.60037
Mean:	2500.	96.2%	2813.43	-2.547	99.6%	1.62528*10^7	-3.006
SD:	99.736		657.187	0.191		2.62375*10^7	1.533

Reporet by Index

```
rptByIndex[d4]
```

Index	Count	T10+ Per	T10+ Avg	T10- Avg	T20+ Per	T20+ Avg	T20-Avg
1	10000	96.8%	3338.85	-2.49971	99.6%	1.10649*10^7	-2.16488
2	10000	95.5%	1999.67	-2.55619	99.6%	1.08306*10^7	-3.46381
3	10000	96.5%	2964.13	-2.57532	99.7%	1.22919*10^7	-4.79882
4	10000	95.8%	2721.04	-2.75824	99.6%	8.31813*10^6	-2.60301
5	10000	96.4%	3034.75	-2.31905	99.6%	3.60863*10^7	-2.32725
Mean:		96.2%	2811.69	-2.542	99.6%	1.57184*10^7	-3.072
SD:			504.598	0.158		1.14772*10^7	1.088

Report of maximum factors for Pos 10

```
 t10HighFac[d4]
```

Base	T10+ Max Factors	Base	T10- Max Factors
79349849016133734643785577 81	3.49131*10^6	219668152827970289682193144354 1	-38.4434
10923790579875709687583085 49	3.49131*10^6	164558256146427091604124869	-19.2217
40416820251932148688144254 29	1.74566*10^6	458297952413679062811844181	-19.2217
126131383382575580748630442290517	872828.	133412925280805298751457928 5	-19.2217
516751062977042569830306501 97	872828.	57838422435688814939616525 65	-19.2217
225082739833962974128241841 49	872828.	60669307135185108339680242 37	-19.2217
19549353960149981100645366 61	872828.	261148867467024273207728336 21	-19.2217
29441700313004032044863125394 77	872828.	24053231081009032646543261 8325	-19.2217
14811374675003447542936159 7125	872828.	12535713081232901590422905 21429	-19.2217
23854875971028992652016416085	872828.	52772407096653498588449471 54261	-19.2217

Report of maximum factors for Pos 20

t20HighFac[d4]

Base	T20+ High Factors	Base	T20- High Factors
77531064614981574401244667630 93	2.53943*10^11	10493540812270639587618269	-34.6381
54923172647710952037729109 3	1.58714*10^10	22309336453143099664453009 0709	-34.6381
84436512121646794643929529029	1.58714*10^10	17749834928813596694089838 93	-17.319
22508273983396297412824184149	1.58714*10^10	33938246610527393639321603 41	-17.319
12700226570425605534790757 3	1.58714*10^10	93955852238911261372871379 89	-17.319
32682439083321259491625529 65	1.58714*10^10	17495119415765581153585940 0789	-17.319
61488601671557315916256290 1	1.58714*10^10	75191260175103476655223319 4581	-17.319
47350896674076041899837670069	7.93571*10^9	28337268700831833338890517	-8.65952
15501029802999181092306032037973	7.93571*10^9	72032270610784925060001 71	-8.65952
77735884095312826598546951917 2949	7.93571*10^9	33408389128308320375595417685	-8.65952

Run 5: 10^30, Sample Size: 50K

```
r5=randSamp[10^30, 10^31, 10000];
d5= mtsProfile[r5];
```

Report by Position

```
rptByPos[d5]
```

Pos,	Count,	T10+Per,	T10+ Q,	T10- Q,	T20+ Per,	T20+ Avg Q,	T20-Avg},
1	2465	96.3%	2200.83	-2.41577	99.8%	6.04732*10^6	-3.24732
2	2580	96.2%	3234.14	-2.63803	99.7%	1.63493*10^7	-2.43549
3	2520	96.5%	2296.78	-2.12967	99.6%	2.58673*10^7	-2.6569
4	2455	95.8%	2166.85	-2.40271	99.8%	1.76122*10^7	-4.69057
5	2395	95.6%	2658.87	-2.58577	99.5%	2.82039*10^7	-5.662
6	2530	96.3%	2678.92	-2.69982	99.6%	2.34787*10^7	-2.81434
7	2370	96.8%	2404.27	-2.71475	99.7%	1.32626*10^7	-3.24732
8	2695	97%	2240.62	-2.66968	99.6%	7.34743*10^6	-3.05051
9	2310	96.7%	3762.7	-2.51336	99.6%	9.21939*10^6	-6.81937
10	2460	95.7%	3244.63	-2.77322	99.3%	1.86245*10^7	-2.36784
11	2515	95.4%	2287.31	-2.51762	99.4%	4.23697*10^6	-3.24732
12	2710	96.2%	2644.31	-2.81494	99.6%	6.76067*10^6	-2.52569
13	2485	96.2%	2505.03	-2.87061	99.6%	7.58022*10^6	-1.62366
14	2620	96.5%	1962.84	-2.22186	99.8%	7.58831*10^6	-1.7319
15	2320	96.8%	2117.3	-2.25855	99.8%	1.21417*10^7	-1.89427
16	2480	96.1%	2167.06	-2.36555	99.8%	2.17868*10^7	-1.35305
17	2545	96.3%	2291.88	-2.80316	99.8%	1.50551*10^7	-1.51542
18	2410	96.1%	2960.88	-2.30154	99.5%	1.5105*10^7	-3.74691
19	2505	96.8%	2477.28	-2.8382	99.8%	5.93468*10^6	-4.97923
20	2630	96.4%	2325.41	-2.52917	99.8%	6.43929*10^6	-3.03083
Mean:	2500.	96.3%	2531.4	-2.553	99.7%	1.34321*10^7	-3.132
SD:	110.834		455.509	0.224		7.31083*10^6	1.443

Report by Position

```
rptByIndex[d5]
```

Index	Count	T10+ Per	T10+ Avg	T10- Avg	T20+ Per	T20+ Avg	T20-Avg
1	10000	96.7%	2674.62	-2.36279	99.6%	9.64288*10^6	-2.77924
2	10000	95.8%	2225.23	-2.59956	99.7%	1.04428*10^7	-3.21124
3	10000	96.6%	2697.01	-2.41676	99.7%	1.49952*10^7	-3.37853
4	10000	95.6%	2285.24	-2.84156	99.6%	1.12248*10^7	-3.97797
5	10000	96.7%	2741.32	-2.45699	99.7%	2.04044*10^7	-2.86528
Mean:		96.3%	2524.68	-2.536	99.7%	1.3342*10^7	-3.242
SD:			248.049	0.192		4.44922*10^6	0.479

Report of maximum factors for Pos 10

```
 t10HighFac[d5]
```

Base	T10+ Max Factors	Base	T10- Max Factors
19558017093731011038611879529731413 3	872828.	19888139082436981502226683249525 3	-19.2217
13760425804315827537148559815218656 5	872828.	39363458103299514644983814176309589	-19.2217
13798910362888138794848597905051938 13	872828.	34950905009352078680686082410862 9	-19.2217
11118562567260334835675118154578056 5	872828.	40739670207331421253174946440557 3	-19.2217
78462084269756731926083620465197781	872828.	75225904389233196869245439991774 5	-19.2217
43409405773980249776356489144773973	872828.	15583100355266095977055649654377 81	-19.2217
32500770586942486212927195076957525	872828.	81324721780794667829245808685540437	-19.2217
10515860438931163039744681499025493	872828.	13682265305718055303466768317741 3397	-19.2217
66994370448869600126205347031687 01	872828.	17311079796362140171016734516261 8197	-19.2217
12597213077770567851522179593129 81	872828.	23599547435882955321405814918949 7173	-19.2217

}

Report of maximum factors for Pos 20

t20HighFac[d5]

Base	T20+ High Factors	Base	T20- High Factors
291200376053219484903520503830474069333	3.17428*10^10	26706142274399656000858218601188181	-34.6381
241700808690330055462722602663705695557	3.17428*10^10	74198288573150902070387693309621	-17.319
103222025453630646113409130574777111189	3.17428*10^10	22898576580054950552475560288647	-17.319
125972130777705678515221795931298 1	3.17428*10^10	16843928339595713889017884419701	-17.319
69224479846161305271831527449237	3.17428*10^10	32119908437909761376484837817915 1189	-17.319
26897936339930441962199175499287 89	1.58714*10^10	11553079404623381548601119820076 37	-8.65952
42032244782155398750980163557737 81	1.58714*10^10	59368849739716407134240160495 71	-8.65952
75720416667802906269516866528700978 5173	1.58714*10^10	96230349325117086891946191321268 7701	-8.65952
44850267031678729842276777891916885	1.58714*10^10	10065180660694195891178322482482 5173	-8.65952
10672304769180317169559940999319 25	1.58714*10^10	10392738542111358695643422075586 2869	-8.65952

Functions

General Functions

hn1[n_]: Uses a formula for generating HN for Element 1. This formula and the next are derived in a separate paper, *Analysis of Number Table Patterns*, also on my website, BandInOne.com.
hn5[n_]: Uses a formula for generating HN for Element 5.
revSeq[n_, Len_:20]: This function generates the reverse sequence (cMTS) for *n*. Twenty terms is the default output; number of terms can be changed with the optional second argument.
getNBR[n_, j_:10]: This function will generate the first 10 terms (default) of a neighborhood.
randSamp[min_, max_, n_:1000]: This function will generate a random sample of 1000 (default) integers in the range {*min, max*). Even numbers are converted to odd numbers.
rnd2[x_]: This function will round a number to two decimal places.
rnd3[x_]: This function will round a number to three decimal places.

```
hn1[n_]:=24n +5
hn5[n_]:=96n+85
revSeq[n_, Len_:20]:=Module[{s, p, L, nextN}, p = n;L = {n};
  nextN[e_]:=Which[Mod[e,3]==0, 4e+1, Mod[e,3]==1, (4e-1)/3,Mod[e,3]==2,(2e-1)/3];
  Do[s = nextN[p]; AppendTo[L, s]; p = s,{i,Len*2}]; Take[Select[L, Mod[#,3 ]!=0&],Len]
  ]
getNBR[n_, j_:10 ]:= Cases[Table[(n*(2^i)-1)/3, {i, 1, j}], _?IntegerQ]
randSamp[min_, max_, n_:1000]:=If[Mod[#,2]==0, #+1, #]&/@RandomInteger[{min, max}, n ]
rnd2[x_]:=Round[x, .01]
rnd2[x_]:=Round[x, .001]
```

Specialized Functions

mtsProfile

The purpose of this function is to document important parameters in cMTS, and to determine if there any correlation between (a) the position of a neighborhood or the index of its first five HN, and (b) the profile of the cMTS of these neighbors. Given a sample of HN (as provide by **randSamp**) the function will take each HN and compute its cMTS. From that cMTS it randomly selects a term index (between 1 and 20) to build a neighborhood. It then identifies the first five HN^C (non-S_0) in that neighborhood and computes their cMTS. For these sequences it records the value of the term in the 5th, 10^{th}, and 20th positions, and compares them with the cBase (the first term of the cMTS). If the cBase is smaller than the distal term, it computes the quotient (distal term/cBase), the factor by which the distal term exceeds the base. If the cBase is higher than the distal term, it computes the quotient (cBase/distal term), the factor by which the cBase exceeds the distal term; these cases are marked as negative. Given that five HN are evaluated for each of the original 10K HN, the total sample size is 50 K per run.

```
mtsProfile[L_List, trm_:30]:=Module[{ser, pos, ind,ind2, len, bas,t,n, ret, val,t5, t10,
t20, cq1,cq2, cq3},ind = 1; len= Length[L];ret={};
  ser = revSeq[#,trm]&/@L;
  While[ind<= len,
   pos =RandomInteger[{1, 20}];
   t[1]=ser[[ind, pos]]; (*get random term of ser[ind]*)
   t[2]= NestList[4#+1&, t[1], 10]; (*builds the neighborhood over term*)
   t[3]= Take[Select[t[2], Mod[#,3]!=0&],5]; (*Selects first 5 non-S0 neighbors*)
   ind2= 1; (*for indexing the HN*)
   While[ind2< 6, (*for each of the 5 neighbors in t[3]this code gets their cMTS, the terms
    at position 5, 10, and 20, and calculates the ratios.  *)
    n[ind2]=revSeq[t[3][[ind2]],21];bas= n[ind2][[1]]; (*get the cBase*)
    t5[ind2]= n[ind2][[5]]; (*gets the 5th term*)
    t10[ind2]= n[ind2][[10]]; (*gets the 10th term*)
    t20[ind2]= n[ind2][[20]];
    If[bas<t5[ind2], cq1[ind2]= N[t5[ind2]/bas], cq1[ind2]=N[bas/-t5[ind2]]];
    If[bas<t10[ind2], cq2[ind2]= N[t10[ind2]/bas], cq2[ind2]=N[bas/-t10[ind2]]];
    If[bas<t20[ind2], cq3[ind2]= N[t20[ind2]/bas], cq3[ind2]=N[bas/-t20[ind2]]];
    AppendTo[ret,{pos, bas, ind2, cq1[ind2],cq2[ind2],cq3[ind2]}];
    ind2=ind2+1
    ];
   ind= ind + 1
   ];
  ret]
```

rptByPos

This function will analyze the data from **mtsProfile** by position. For each of the 20 positions it will report:

Position number: The term index that is randomly chosen; (between 1 and 20).

Count of items: Term position is chosen randomly which leads to variable counts, but the total for all positions will be 50K.

***T*05 + Per**: The percent of term 5 values that were greater than the cBase of the cMTS.

***T*05 + Avg Q**: The mean quotient of (distal term/cBase).

***T*5 − Avg Q**: The mean quotient of (cBase/distal term).

***T*10 + Per**: The percent of term 10 values that were greater than the cBase of the cMTS.

***T*10 + Avg Q**: The mean quotient of (distal term/cBase).

***T*10 − Avg Q**: The mean quotient of (cBase/distal term).

***T*20 + Per**: The percent of term 20 values that were greater than the cBase of the cMTS.

***T*20 + Avg Q**: The mean quotient of (distal term/cBase).

***T*20 − Avg Q**: The mean quotient of (cBase/distal term).

(See next page.)

```
rptByPos[L_List]:=Module[{p, rpt, n,
   c05,q05p,q05pa, q05n,q05na, a05pc=0, a05pm=0, a05nc=0,a05nm=0,perc05p,per05avg={},
   a05pcD={}, a05pmD={}, a05ncD={},a05nmD={},
   c10,q10p,q10pa, q10n,q10na, a10pc=0, a10pm=0, a10nc=0,a10nm=0,perc10p,per10avg={},
   a10pcD={}, a10pmD={}, a10ncD={},a10nmD={},
   c20,q20p,q20pa, q20n,q20na, a20pc=0, a20pm=0, a20nc=0,a20nm=0, perc20p,per20avg={},
    a20pcD={}, a20pmD={}, a20ncD={},a20nmD={}},
  rpt= {};n=1; (*This While organizes data by position; There are 20 possible positions*)
  While[n<21,
   p[n]=Select[L, #[[1]] == n&];
   n = n+1 ];  n = 1;
  While[n<21,
   q05p[n]=Select[p[n], #[[4]] >0&][[All,4]];If[q05p[n]=={},q05p[n]=Null];
   If[Length[q05p[n]]>0,q05pa[n]= Mean[q05p[n]],q05pa[n]=Null];
   q05n[n]=Select[p[n], #[[4]] <0&][[All,4]];If[q05n[n]=={},q05n[n]=Null];
   If[Length[q05n[n]]>0,q05na[n]= Mean[q05n[n]],q05na[n]=Null];
   c05[n]= Length[q05p[n]]+ Length[q05n[n]];
   perc05p=rnd3[Length[q05p[n]]/c05[n]];AppendTo[per05avg, perc05p];
   q10p[n]=Select[p[n], #[[5]] >0&][[All,5]];If[q10p[n]=={},q10p[n]=Null];
   If[Length[q10p[n]]>0,q10pa[n]= Mean[q10p[n]],q10pa[n]=Null];
   q10n[n]=Select[p[n], #[[5]] <0&][[All,5]];If[q10n[n]=={},q10n[n]=Null];
   If[Length[q10n[n]]>0,q10na[n]= Mean[q10n[n]],q10na[n]=Null];
   c10[n]= Length[q10p[n]]+ Length[q10n[n]];
   perc10p=rnd3[Length[q10p[n]]/c10[n]];AppendTo[per10avg, perc10p];
   q20p[n]=Select[p[n], #[[6]] >0&][[All,6]];If[q20p[n]=={},q20p[n]=Null];
   If[Length[q20p[n]]>0,q20pa[n]= Mean[q20p[n]],q20pa[n]=Null];
   q20n[n]=Select[p[n], #[[6]] <0&][[All,6]];If[q20n[n]=={},q20n[n]=Null];
   If[Length[q20n[n]]>0,q20na[n]= Mean[q20n[n]],q20na[n]=Null];
   c20[n]= Length[q20p[n]]+ Length[q20n[n]];
   perc20p=rnd3[Length[q20p[n]]/c20[n]];AppendTo[per20avg, perc20p];
   AppendTo[rpt,{n,c10[n], PercentForm[perc05p],q05pa[n], q05na[n], PercentForm[perc10p],
    q10pa[n], q10na[n], PercentForm[perc20p],q20pa[n], q20na[n]}];
   n = n+1];n=1; (*The following While is for calculating averages and SD.*)
  While[n<21,
   AppendTo[a05pcD,c05[n]]; a05pc= a05pc+ c05[n];
   AppendTo[a05pmD, q05pa[n]];a05pm= a05pm+q05pa[n];
   AppendTo[a05ncD,Length[q05n[n]]]; a05nc= a05nc +Length[q05n[n]];
   AppendTo[a05nmD, q05na[n]];a05nm= a05nm + q05na[n];
   AppendTo[a10pcD,c10[n]]; a10pc= a10pc+ c10[n];
   AppendTo[a10pmD, q10pa[n]];a10pm= a10pm+q10pa[n];
   AppendTo[a10ncD,Length[q10n[n]]]; a10nc= a10nc +Length[q10n[n]];
   AppendTo[a10nmD, q10na[n]];a10nm= a10nm + q10na[n];
   AppendTo[a20pcD,Length[q20p[n]]];a20pc= a20pc+ Length[q20p[n]];
   AppendTo[a20pmD,q20pa[n]]; a20pm= a20pm+q20pa[n];
   AppendTo[a20ncD,Length[q20n[n]]];a20nc= a20nc +Length[q20n[n]];
   AppendTo[a20nmD,q20na[n]];a20nm= a20nm + q20na[n];
   n = n+1];
  AppendTo[rpt,{"Mean:",rnd2[a10pc/20],PercentForm[rnd3[Mean[per05avg]]],
   rnd2[Mean[a05pmD]], rnd2[a05nm/20], PercentForm[rnd3[Mean[per10avg]]],
   rnd2[Mean[a10pmD]], rnd2[a10nm/20], PercentForm[rnd3[Mean[per20avg]]], rnd2[a20pm/20],
   rnd2[a20nm/20]}];
  AppendTo[rpt,{"  SD:",rnd2[StandardDeviation[a10pcD]], {},
   rnd2[StandardDeviation[a05pmD]], rnd2[StandardDeviation[a05nmD]],{},
   rnd2[StandardDeviation[a10pmD]], rnd2[StandardDeviation[a10nmD]],{},
   rnd2[StandardDeviation[a20pmD]], rnd2[StandardDeviation[a20nmD]]}];
  TableForm[rpt,TableHeadings->{None, {"Pos", "Count ", "T5+ Per","T5+ Avg Q", "T5- Avg Q",
       "T10+ Per","T10+ Avg Q", "T10- Avg Q",  "T20+ Per","T20+ Avg Q", "T20-Avg"}}]
    ]
```

rptByIndex

This function does the same type of analysis as rptByPosition, except that it organizes the data by HN index.

```
rptByIndex[L_List]:=Module[{i, rpt, n,
   t05,q05p,q05pa, q05n,q05na, a05pc=0,a05nc=0,a05pm=0,a05nm=0,t05per={},a05pcD={},
a05pmD={}, a05ncD={},a05nmD={},
   t10,q10p,q10pa, q10n,q10na, a10pc=0,a10nc=0,a10pm=0,a10nm=0,t10per={},a10pcD={},
a10pmD={}, a10ncD={},a10nmD={},
   t20,q20p,q20pa, q20n,q20na, a20pc=0,a20nc=0,a20pm=0,a20nm=0,t20per={},a20pcD={},
a20pmD={}, a20ncD={},a20nmD={}},
  rpt= {};n=1;   (*This While organizes the data by index; There are 5 possible indices*)
While[n<6,
   i[n]=Select[L, #[[3]] == n&];
   n = n+1];   n = 1;
While[n<6,q05p[n]=Select[i[n], #[[4]] >0&][[All,4]];If[q05p[n]=={},q05p[n]=Null];
 (*Select the +5 set*)
If[Length[q05p[n]]>0,q05pa[n]= Mean[q05p[n]],q05pa[n]=Null]; (*Gets their avg*)
q05n[n]=Select[i[n], #[[4]] <0&][[All,4]];If[q05n[n]=={},q05n[n]=Null];
(*Select the -5 set*)
If[Length[q05n[n]]>0,q05na[n]= Mean[q05n[n]],q05na[n]=Null];   (*Gets their avg*)
q10p[n]=Select[i[n], #[[5]] >0&][[All,5]];If[q10p[n]=={},q10p[n]=Null];
  If[Length[q10p[n]]>0,q10pa[n]= Mean[q10p[n]],q10pa[n]=Null];
q10n[n]=Select[i[n], #[[5]] <0&][[All,5]];If[q10n[n]=={},q10n[n]=Null];
  If[Length[q10n[n]]>0,q10na[n]= Mean[q10n[n]],q10na[n]=Null];
q20p[n]=Select[i[n], #[[6]] >0&][[All,6]];If[q20p[n]=={},q20p[n]=Null];
  If[Length[q20p[n]]>0,q20pa[n]= Mean[q20p[n]],q20pa[n]=Null];
q20n[n]=Select[i[n], #[[6]] <0&][[All,6]];If[q20n[n]=={},q20n[n]=Null];
  If[Length[q20n[n]]>0,q20na[n]= Mean[q20n[n]],q20na[n]=Null];
t05=  Length[q05p[n]]+Length[q05n[n]];AppendTo[t05per,rnd2[Length[q05p[n]]/t05] ];
t10=  Length[q10p[n]]+Length[q10n[n]];AppendTo[t10per,rnd2[Length[q10p[n]]/t10] ];
t20=  Length[q20p[n]]+Length[q20n[n]];AppendTo[t20per,rnd2[Length[q20p[n]]/t20] ];
AppendTo[rpt,{n, t10,PercentForm[rnd2[Length[q05p[n]]/t05]], q05pa[n],  q05na[n],
PercentForm[rnd2[Length[q10p[n]]/t10]], q10pa[n],  q10na[n],
PercentForm[rnd2[Length[q20p[n]]/t20]],q20pa[n],  q20na[n]}];
 n = n+1 ];   n=1;
 While[n<6,
   AppendTo[a05pcD,Length[q05p[n]]]; a05pc= a05pc+ Length[q05p[n]];
   AppendTo[a05pmD, q05pa[n]];a05pm= a05pm+q05pa[n];
   AppendTo[a05ncD,Length[q05n[n]]]; a05nc= a05nc +Length[q05n[n]];
   AppendTo[a05nmD, q05na[n]];a05nm= a05nm + q05na[n];
   AppendTo[a10pcD,Length[q10p[n]]]; a10pc= a10pc+ Length[q10p[n]];
   AppendTo[a10pmD, q10pa[n]];a10pm= a10pm+q10pa[n];
   AppendTo[a10ncD,Length[q10n[n]]]; a10nc= a10nc +Length[q10n[n]];
   AppendTo[a10nmD, q10na[n]];a10nm= a10nm + q10na[n];
   AppendTo[a20pcD,Length[q20p[n]]];a20pc= a20pc+ Length[q20p[n]];
   AppendTo[a20pmD,q20pa[n]]; a20pm= a20pm+q20pa[n];
   AppendTo[a20ncD,Length[q20n[n]]];a20nc= a20nc +Length[q20n[n]];
   AppendTo[a20nmD,q20na[n]];a20nm= a20nm + q20na[n];
   n=n+1; ];
AppendTo[rpt,{"Mean: ", {},PercentForm[rnd2[Mean[t05per]]], rnd2[a05pm/5], rnd2[a05nm/5],
  PercentForm[rnd2[Mean[t10per]]], rnd2[a10pm/5], rnd2[a10nm/5],
  PercentForm[rnd2[Mean[t20per]]], rnd2[a20pm/5], rnd2[a20nm/5]}];
AppendTo[rpt,{"  SD:",{},{}, rnd2[StandardDeviation[a05pmD]],
  rnd2[StandardDeviation[a05nmD]],{}, rnd2[StandardDeviation[a10pmD]],
  rnd2[StandardDeviation[a10nmD]],{}, rnd2[StandardDeviation[a20pmD]],
  rnd2[StandardDeviation[a20nmD]]}];
TableForm[rpt,TableHeadings->{None, {"Index","Count","T5+ Per",  "T5+ Avg", "T5- Avg",
      "T10+ Per",  "T10+ Avg", "T10- Avg", "T20+ Per", "T20+ Avg", "T20-Avg"}}]
  ]
```

High Factor reports

These two reports lake the output from **mtsProfile** and select the 10 (default) highest factors for T10+ and $T10-$. or $T20+$ and $T20-$. You can request a deeper list using the second argument

```
t10HighFac[L_List, Len_:10 ]:=Module[{a, b, c}, a= Take[ReverseSortBy[Select[L, #[[4]]>
0&][[All, {2, 4}]], Last], Len];
  b=Take[SortBy[Select[L, #[[4]]< 0&][[All, {2, 4}]], Last], Len];
  c=Map[Flatten,Transpose[{a, b}],2];
  TableForm[c, TableHeadings->{None, {"Base", "T10+ High Factors","Base", "T10- High
Factors"}}]
  ]
```

```
t20HighFac[L_List, Len_:10 ]:=Module[{a, b, c}, a= Take[ReverseSortBy[Select[L, #[[5]]>
0&], Last][[All, {2, 5}]], Len];
  b=Take[SortBy[Select[L, #[[5]]< 0&], Last][[All, {2, 5}]], Len];
  c=Map[Flatten,Transpose[{a, b}],2];
  TableForm[c, TableHeadings->{None, {"Base", "T20+ High Factors","Base", "T20- High
Factors"}}]
  ]
```