

# Finite Summation of Integer Powers $x^p$ , Part 2

Assad Ebrahim\*

1/30/2010

(Rev. 0f, April 4, 2010)

## Abstract

We solve the general case of the finite-summation-of-integer-powers problem  $S_p(N) = \sum_{k=1}^N k^p$  for arbitrary  $p$  using recurrence relations to obtain a  $p$ -th order recurrence relation which can be used iteratively to obtain the closed form formula for any given  $p$ .

We include the Maxima source code for computing the closed form formulas for any given  $p$  using the  $p$ -th order recurrence solution obtained in this paper.

The motivated development of the recurrence relation approach, and illustration for the special cases  $p = 1, 2, 3, 4$ , are given in Part 1 [Ebr10] of this three-part paper. A direct (non-iterative) solution using matrix methods is given in Part 3 [EO10].

**Finding**  $S_p(N) = \sum_{k=1}^N k^p$

We desire a general formula that solves the finite-summation-of-integer-powers problem for any positive integer power  $p$ :

$$S_p(N) := \sum_{k=1}^N k^p, \text{ (where } p \in \mathbb{N} = \{0, 1, 2, \dots\}; N \geq 0). \quad (1)$$

We proceed using the method of recurrence relations that was illustrated for small  $p$  in Part 1 of this paper [Ebr10] and that we prove inductively here.

## Outline

We obtain a simple first-order recurrence relation (2) and a more complicated recurrence relation (5) that involves the coefficients of a lower order solution to (1). A sequence of manipulations followed by substitution of the first recurrence into the second yields the recurrence relation solution (10). Induction on  $p$  shows that this is indeed the solution we seek, and that the closed form solution is a

---

\*Assad Ebrahim received his M.Sc. in Applied Mathematics from the University of Washington in Seattle and his B.A. in Mathematics from Swarthmore College in Philadelphia. He is Director of Engineering and Operations at BioSonics, Inc., a company specializing in the design and manufacture of sonar systems and software for the detection, analysis, and classification of sonar signals applied to fisheries research, environmental monitoring, and homeland defense. (<http://www.biosonicsinc.com>) He is also the Founder of Mathematical Science & Technologies (<http://www.mathscitech.org>), an organization involved in the application of mathematics, technology, and applied science to industry, and in the enhancement of curricula for STEM education (Science, Technology, Engineering, and Mathematics). Contact Information: [assad.ebrahim@mathscitech.org](mailto:assad.ebrahim@mathscitech.org).

polynomial of degree  $p + 1$  with rational coefficients and no constant term. The recurrence relation (10) allows the closed form expressions to (1) to be computed iteratively for any  $p$ .  $\square$  A direct (non-iterative) matrix method for computing the closed form solutions is given in Part 3 [EO10] of this paper.

Proceeding now to the solution:

### Deriving the Solution

**Claim 1** *The solution to (1) is a polynomial of degree  $p + 1$  with rational coefficients and no constant term.*

In particular we shall find an expression (10) for this polynomial that allows a closed form for (1) to be computed for any given  $p$  and all  $N$ .

**Proof** (By Induction) Base case  $p = 0$ : the solution is immediate:  $S_0(N) = \sum_{k=1}^N 1 = N$ . This is a polynomial of degree 1 with rational coefficients and no constant term.

For the inductive case, suppose the claim to be true for powers  $1, 2, \dots, p-1$ . We shall show that it is then true for case  $p$ .

Obtain the following recurrence directly from the problem statement (1):

$$S_p(N) = S_p(N - 1) + N^p. \quad (2)$$

This is a first recurrence.

Obtain a second recurrence from the problem statement (1) as follows: we have supposed in our induction hypothesis that the closed form solutions for the lower order problems are polynomials of a certain form. So we may write the  $p - 1$  solution as:

$$S_{p-1}(N) = \alpha_p N^p + \sum_{j=1}^{p-1} \alpha_j N^j. \quad (p \geq 1; N \geq 0) \quad (3)$$

In writing (3), the highest order term,  $N^p$ , and its coefficient  $\alpha_p$ , are emphasized.

Change variables in (3) from  $N$  to  $K$ , and solve for the highest order term  $K^p$ :

$$K^p = \frac{1}{\alpha_p} \left[ S_{p-1}(K) - \sum_{j=1}^{p-1} \alpha_j K^j \right]. \quad (4)$$

Substituting (4) into (1) gives the desired second recurrence, which we break into two summations for separate treatment:

$$\begin{aligned} S_p(N) &= \sum_{K=1}^N \frac{1}{\alpha_p} \left[ S_{p-1}(K) - \sum_{j=1}^{p-1} \alpha_j K^j \right] \\ &= \frac{1}{\alpha_p} \sum_{K=1}^N S_{p-1}(K) - \frac{1}{\alpha_p} \sum_{K=1}^N \sum_{j=1}^{p-1} \alpha_j K^j. \end{aligned} \quad (5)$$

Considering each summation in (5) separately:

**First summation in (5)...**

Expand the first summation and gather the terms by like  $(p-1)$ -powers of  $K$ :

$$\begin{aligned}
\sum_{k=1}^N S_{p-1}(K) &= S_{p-1}(1) + S_{p-1}(2) + S_{p-1}(3) + \dots + S_{p-1}(N) \\
&= 1^{p-1} + (1^{p-1} + 2^{p-1}) + (1^{p-1} + 2^{p-1} + 3^{p-1}) + \dots + (1^{p-1} + 2^{p-1} + 3^{p-1} + \dots + N^{p-1}) \\
&= N(1^{p-1}) + (N-1)(2^{p-1}) + \dots + 2(N-1)^{p-1} + 1(N^{p-1}) \\
&= \sum_{k=0}^{N-1} (N-K)(K+1)^{p-1}. \tag{6}
\end{aligned}$$

Expand the binomial power in (6) using the binomial formula:

$$\begin{aligned}
&= \sum_{k=0}^{N-1} (N-K) \sum_{j=0}^{p-1} \binom{p-1}{j} K^j \\
&= \sum_{k=0}^{N-1} \sum_{j=0}^{p-1} (N-K) \binom{p-1}{j} K^j \\
&= \sum_{k=0}^{N-1} \sum_{j=0}^{p-1} \binom{p-1}{j} [NK^j - K^{j+1}].
\end{aligned}$$

Interchange the order of summation:

$$= \sum_{j=0}^{p-1} \binom{p-1}{j} \sum_{k=0}^{N-1} [NK^j - K^{j+1}].$$

Pull the  $K=0$  term out of both summations. NOTE:  $0^0 = 1$  — see footnote (next page)<sup>1</sup>

$$\begin{aligned}
&= N + \sum_{j=0}^{p-1} \binom{p-1}{j} \sum_{k=1}^{N-1} [NK^j - K^{j+1}] \\
&= N + \sum_{j=0}^{p-1} \binom{p-1}{j} \left[ \sum_{k=1}^{N-1} NK^j - \sum_{k=1}^{N-1} K^{j+1} \right].
\end{aligned}$$

Recognize the appearance of the finite sum of integer powers, but of lower order:

$$= N + \sum_{j=0}^{p-1} \binom{p-1}{j} NS_j(N-1) - \sum_{j=0}^{p-1} \binom{p-1}{j} S_{j+1}(N-1).$$

Shifting indices in the second summation, see [GKP], [Knu] for index manipulation techniques:

$$= N + \sum_{j=0}^{p-1} \binom{p-1}{j} NS_j(N-1) - \sum_{j=1}^p \binom{p-1}{j-1} S_j(N-1).$$

Peel off 0-th term from first sum and  $p$ -th term from second sum to combine the rest:

$$\begin{aligned}
&= N + N(N-1) - S_p(N-1) + \sum_{j=1}^{p-1} S_j(N-1) \left[ N \binom{p-1}{j} - \binom{p-1}{j-1} \right] \\
&= N^2 - S_p(N-1) + \sum_{j=1}^{p-1} S_j(N-1) \left[ N \binom{p-1}{j} - \binom{p-1}{j-1} \right]. \tag{7}
\end{aligned}$$

**Second summation in (5)...**

The second summation term in (5) is a double summation. Recognize that the finite sum of  $K^j$  summed over  $K$  is in fact  $S_j(N)$ . So interchange the order of summation and proceed:

$$\begin{aligned} \sum_{k=1}^N \sum_{j=1}^{p-1} \alpha_j K^j &= \sum_{j=1}^{p-1} \alpha_j \left( \sum_{k=1}^N k^j \right) \\ &= \sum_{j=1}^{p-1} \alpha_j S_j(N). \end{aligned} \quad (8)$$

**Arriving at the final form of the second recurrence...**

Combining (7) and (8) into the second recurrence (5) gives the simplified second recurrence:

$$S_p(N) = \frac{1}{\alpha_p} \left\{ N^2 - S_p(N-1) + \sum_{j=1}^{p-1} C_j(N) S_j(N-1) - \sum_{j=1}^{p-1} \alpha_j S_j(N) \right\}, \quad (9)$$

where

$$C_j(N) = \left[ \binom{p-1}{j} N - \binom{p-1}{j-1} \right]$$

is a linear polynomial in  $N$  with binomial coefficients.

Substituting  $S_p(N-1)$  from the first recurrence (2) into (9) yields:

$$S_p(N) = \frac{1}{\alpha_p} \left\{ N^2 - S_p(N) + N^p + \sum_{j=1}^{p-1} C_j(N) S_j(N-1) - \sum_{j=1}^{p-1} \alpha_j S_j(N) \right\}.$$

Collecting the  $S_p(N)$  terms and simplifying gives a  $p$ -th order recurrence relation that supplies a closed form expression for  $S_p(N)$  in terms of lower order summations:

$$S_p(N) = \frac{1}{1 + \alpha_p} \left\{ N^2 + N^p + \sum_{j=1}^{p-1} C_j(N) S_j(N-1) - \sum_{j=1}^{p-1} \alpha_j S_j(N) \right\}, \quad (10)$$

---

<sup>1</sup>Peel off the  $K = 0$  term of the inner summation to get:  $N0^j - 0^{j+1}$ . Peeling this out of the outer summation requires considering the value of the expression when summed over *all* values of  $j$ . Now, 0 raised to any *positive* power is 0, so we can dispel the case  $j > 0$ . This leaves:

$$\binom{p-1}{0} (N0^0 - 0^1)$$

What is  $0^0$ ? A decision must be made: it is either 0 or 1. Indeterminacy is not an option, since the situation is real and is required to continue the simplification. This step in the derivation of (7) from (6) relies on the empirical fact that  $0^0 = 1$ .

**Definition 2 (Empirical)**  $0^0 = 1$  for  $k^j$ , where  $k, j$  are discrete variables.

(The empirical basis for this definition is discussed in Appendix C.) With this definition, the value of the  $K = 0$  term summed over all values of  $j$  is  $N$ , and we proceed with the simplification toward (7).

where the  $\alpha_j$  are the coefficients from the closed form solution of  $S_{p-1}(N)$ , and

$$C_j = \left[ N \binom{p-1}{j} - \binom{p-1}{j-1} \right].$$

Inspection shows that the general solution (10) is a polynomial in  $N$  that satisfies all three elements of Claim 1:

1. it has degree  $p + 1$ :  $S_{p-1}(N - 1)$  is a  $p$ -th order polynomial multiplied by the linear polynomial  $C_p - 1(N)$ ,
2. it has no constant term
3. it has rational coefficients: the coefficients are sums and products of coefficients of lower order solutions (which are rational by the inductive hypothesis) and binomial coefficients (positive integers).

The Claim is therefore shown by induction.  $\square$

**Remarks** We have in (10) a recurrence formula for the finite-summation-of-integer-powers problem that can be used to derive closed form expressions for any given  $p$ .

**Listing of Closed Form Solutions** Figure 1 shows the first 10 closed form expressions computed iteratively using Maxima. Maxima source code is provided in Appendix A.

## Conclusion

### Summary

We have developed a  $p$ -th order recurrence relation (10) that, when used iteratively, gives the general closed form formula for the finite summation  $\sum_{k=1}^N k^p$  for any given  $p$ .

Computing closed form formulas for specific  $p$  from this  $p$ -th order recurrence is made substantially easier by a computer package, whether this is a symbolical algebra package such as Maxima, that gets through the tangle of algebra that arises out of what is effectively a chain of substitutions, or a programming language such as Ruby, that has flexible language structures allowing an iteration such as this to be coded up easily.

Appendix A provides Maxima source code that uses (10) to iteratively compute the closed form expression for  $S_p(N)$  for any given  $p$ .

### Next Steps

Can we do better? Specifically, can we find a closed form expression for solution that can be obtained directly, i.e. without requiring iteration? What relations hold between and among the  $S_p(N)$ ? These matters are taken up in Part 3 of this paper [EO10]. In particular, Part 3 uses matrix methods to obtain the closed form expressions shown in Figure 1 directly and without requiring iteration, by solving a linear system.

```
(%i6) for i:1 thru 10 do print(SpN(i));
```

$$s_1(n) := \frac{n^2 + n}{2}$$

$$s_2(n) := \frac{2n^3 + 3n^2 + n}{6}$$

$$s_3(n) := \frac{n^4 + 2n^3 + n^2}{4}$$

$$s_4(n) := \frac{6n^5 + 15n^4 + 10n^3 - n}{30}$$

$$s_5(n) := \frac{2n^6 + 6n^5 + 5n^4 - n^2}{12}$$

$$s_6(n) := \frac{6n^7 + 21n^6 + 21n^5 - 7n^3 + n}{42}$$

$$s_7(n) := \frac{3n^8 + 12n^7 + 14n^6 - 7n^4 + 2n^2}{24}$$

$$s_8(n) := \frac{10n^9 + 45n^8 + 60n^7 - 42n^5 + 20n^3 - 3n}{90}$$

$$s_9(n) := \frac{2n^{10} + 10n^9 + 15n^8 - 14n^6 + 10n^4 - 3n^2}{20}$$

$$s_{10}(n) := \frac{6n^{11} + 33n^{10} + 55n^9 - 66n^7 + 66n^5 - 33n^3 + 5n}{66}$$

Figure 1: Closed form expressions for  $S_p(N) = \sum_{k=1}^N k^p$ , for  $p = 1, 2, \dots, 10$ , computed using (10) in computer algebra system, Maxima. Source code for  $S_p(N)$  is given in Appendix A.

## Appendix A: Maxima Source Code for Computing $S_p(N) := \sum_{k=1}^N k^p$

This Appendix gives Maxima source code for computing closed form expressions for the finite-summation-of-integer-powers problem (1) using the  $p$ -th order recurrence relation (10). Appendix B shows how both Maxima and Ruby can be used for cross-checking both the derivation and the correctness of the implementation.

```

/* MAXIMA SOURCE CODE
  Author: Assad Ebrahim, (assad.ebrahim@mathscitech.org)
  License: LGPL (Lesser GPL)
  Copyright (c) 2010, Assad Ebrahim.

  Iteratively Computing Finite Sum of P-th Powers of Integers
  using the p-th order recurrence (10) from the paper:
  "Finite Summation of Integer Powers, Part 2", Assad Ebrahim, Jan 2010
  Paper at: http://mathscitech.org/papers/ebrahim-sum-powers-2.pdf
  Source code at: http://mathscitech.org/articles/downloads#source

/* *****
  USAGE INSTRUCTIONS: from within wxMaxima, run the following commands:

  load("c:\\tmp\\sumkp_recurrence.mac"); [replace with path to downloaded file]
  SpN(p);                               [positive integer p: 1,2,...]
  SpN_pretty(p);

  NOTES: - SpN(p) returns the closed form formula for the finite summation
          of p-th powers of integers
          - SpN_pretty(p) produces a prettified output as well.
  */

/* *****
  PUBLIC Function: Iteratively computes S_p(N)
  Input: positive integer p: 1,2,...
  Return: formula for S_p(N) */

SpN(p) := (
  for i:1 thru p-1 do get_next_SpN(i),
  get_next_SpN(p) /* return */
)$

/* *****
  PUBLIC Function: Iteratively computes S_p(N). Pretty prints solution.
  Input: positive integer p: 1,2,...
  Return: formula for S_p(N) */

SpN_pretty(p) := (
  print(sum(k^p,k,1,n)," = ",SpN(p)) /* pretty print */
)$

/* *****
  PRIVATE FUNCTIONS
  ***** */

```

```

/* *****
CORE COMPUTATION: p-th ORDER RECURRENCE
-----
PRIVATE Function: Computes p-th solution using p-th order recurrence relation
of Equation (1-SOL), and prior solutions.
Solution in: "Finite Summation of Integer Powers, Part 2", A. Ebrahim, 1/2010
Paper at: http://mathscitech.org/papers/ebrahim-sum-powers-2.pdf */

get_next_SpN(p) := block([a,sum1,sum2],

s[0](n) := n, /* initial value */
c(p,j,n) := n*binomial(p-1,j) - binomial(p-1,j-1), /* definition */

a[p-1] : getcoeffs(s,p-1), /* get prior solution's coefficients */

/* compute the two summations in Equation (1-SOL) */
sum1 : ratsimp(sum(c(p,k,n)*s[k](n-1),k,1,p-1)),
sum2 : ratsimp(sum(a[p-1][k+1]*s[k](n),k,1,p-1)),

/* compute and assign solution (1-SOL) */
define(s[p](n),ratsimp((1/(1+a[p-1][p+1]))*(n^2 + n^p + sum1 - sum2)))
)$

/* *****
PRIVATE Function: get coefficients of prior solution */

getcoeffs(s,p) := (
v : makelist(ratcoeff(s[p](n),n,j),j,0,p+1) /* jth coeff. <--> n^j power*/
)$

/* *****
Package complete */
print("Loading completed successfully.");
print("Commands: SpN(p) or SpN_pretty(p).");

/*
Revision History:
2/24/2010 - 002 - AKE - revised comments, minor streamlining
2/23/2010 - 001 - AKE - initial */

```

## Appendix B: Using computing software to compute and check the derivation (10)

### Cross-Checking Derivation of the General Formula Using Maxima and Ruby

We can check the derivation using computing software, as follows:

- Use Maxima to obtain the closed form expression for  $S_5(N)$  (Figure 2). Now, you could determine the  $C_j(N)$  by hand, e.g.

$$C_1 = \binom{4}{1}N - \binom{4}{0} = 4N - 1$$

$$C_2 = \binom{4}{2}N - \binom{4}{1} = 6N - 4$$

$$C_3 = \binom{4}{3}N - \binom{4}{2} = 4N - 6$$

$$C_4 = \binom{4}{4}N - \binom{4}{3} = N - 4,$$

but why bother? Maxima has the facility to evaluate binomial coefficients, so just code up the general definition:

```
c(p, j, n) := n*binom(p-1, j) - binom(p-1, j-1);
```

$$S_5(N) = \frac{1}{12} [2N^6 + 6N^5 + 5N^4 - N^2]$$

- Use Maxima to evaluate this at, say,  $N = 10$  (Figure 3)

$$S_5(10) = 220,825$$

- Use Ruby to perform the brute force summation  $S_5(10) = \sum_{k=1}^{10} k^5$  (Figure 4). This yields the same value: 220,825.

Observe that the two computations of  $S_5(10)$  match. One can check at various values of  $N$  and convince oneself that the derivation is correct. Note: Being convinced of the correctness of a derivation through computing values is not the same thing as a *proof* of correctness. It is, however, important to catch errors, both typographical and in implementation.

## Appendix C: Why $0^0 = 1$

A key step in the derivation of (7) from (6) relies on the fact that  $0^0 = 1$ . Recall, we peeled off the  $K = 0$  term of the inner summation to get:  $N0^j - 0^{j+1}$ . Peeling this out of the outer summation requires considering the expression for *all*  $j$ . Now, 0 raised to any *positive* power is 0, so we can dispel the case of  $j > 0$ . But what is  $0^0$ ? A decision must be made: it is either 0 or 1. Indeterminacy is not an options, since the situation is real and is required to continue the simplification.

```
(%i1) load("c:\\tmp\\sumkp_recurrence.mac");

(%i3) SpN_pretty(5);
```

$$\sum_{k=1}^n k^5 = s_5(n) := \frac{2n^6 + 6n^5 + 5n^4 - n^2}{12}$$

```
(%o3) s_5(n) := \frac{2n^6 + 6n^5 + 5n^4 - n^2}{12}
```

Figure 2: Obtaining closed form for  $S_5(N)$  using Symbolic Simplification in Maxima

```
(%i18) s5(N):=(1/12)*(2*N^6 + 6*N^5 + 5*N^4 - N^2);

(%i19) s5(10);
(%o19) 220825
```

Figure 3: Evaluation of  $S_5(10)$  in Maxima

**Aside: Why  $0^0 = 1$ ?** The question of what value  $0^0$  should evaluate to has been discussed since the time of Euler (1700s). There are three choices: 1, 0, or “indeterminate”, i.e. throw an error.<sup>2</sup>

The difficulty arises when considering  $x^y$  as a function of two continuous variables. If we take  $x, y \neq 0$ , and then consider  $y \rightarrow 0$ , we have  $x^y \rightarrow 1$ . But if we consider  $x \rightarrow 0$ , then we have  $x^y \rightarrow 0$ . Clearly,  $x^y$ , as a function of two variables, is discontinuous at  $(0, 0)$ . This is the reason for L’Hopital’s rule in Calculus that requires treating limits that tend to  $0^0$  by special methods. Euler and Knuth both argued strongly that  $0^0 = 1$ , based on the proof of the binomial theorem, and on its appearance in discrete mathematics (double summations).

<sup>2</sup>Further discussion of  $0^0$  is at:  
<http://mathforum.org/dr.math/faq/faq.0.to.0.power.html>

```
irb(main):001:0> s5=0;
irb(main):002:0* 1.upto(10) do !k! s5+=k**5 end;
irb(main):003:0* s5
=> 220825
```

Figure 4: Brute force summation  $S_5(10) = \sum_{k=1}^{10} k^5$  in Ruby (irb)

Though “majority opinion” is not an acceptable reason, it is instructive to consider how various computing platforms treat  $0^0$ : Google, Ruby, R, Octave all evaluate  $0^0$  as 1. Microsoft’s Calculator treats  $0^0$  as 1. Maxima and Microsoft’s Excel (2000), treat  $0^0$  as indeterminate – i.e., they throw an error. Some older hand calculators (TI-32) treat  $0^0$  as 0.

There should be a reason for all mathematical decisions. Often, where there is more than one choice, or where a definitional decision is required, the decision is made based upon consistency and/or generality. Indeed, this is the reason why  $(-1)(-1) = 1$  — to be consistent with the distributive law of multiplication over addition, extended from the positive whole numbers  $\mathbb{N}$  to the integers  $\mathbb{Z}$ .

In the finite summation of integer powers, we have a problem with a real, tangible result (a finite sum). Considering the alternatives  $0^0 = 0$  and  $0^0 = 1$  and checking the resulting formula against the actual summation lends support for the definition  $0^0 = 1$  (at least in discrete mathematics):

**Definition 3 (Empirical)**  $0^0 = 1$  for  $k^j$ , where  $k, j$  are discrete variables.

Experimentation in Ruby shows that the assumption of  $0^0 = 0$  leads to the calculation of  $S_5(10)$  to be off by a linear constant  $N$ , while  $0^0 = 1$  leads to the exact formula for  $S_5(N)$ .

## Acknowledgements

I thank Carol Ouellette for reading an early draft of this paper and for many thoughtful comments and suggestions.

## References

- [Ebr10] Assad Ebrahim. Finite summation of integer powers  $x^p$ , part 1. January 2010.
- [EO10] Assad Ebrahim and Carol Ouellette. Finite summation of integer powers  $x^p$ , part 3. February 2010.
- [GKP] Ron Graham, Donald Knuth, and Oren Patashnik. *Concrete Mathematics: A Foundation for Computer Science*. Addison Wesley.
- [Knu] Donald Knuth. *The Art of Computer Programming (3 Volumes)*.