ISSN 1547-4771, Physics of Particles and Nuclei Letters, 2020, Vol. 17, No. 5, pp. 760–762. © Pleiades Publishing, Ltd., 2020.

PHYSICS OF ELEMENTARY PARTICLES AND ATOMIC NUCLEI. THEORY

Solving Recurrence Relations for Radial Pullback of Cubic Interaction in AdS_{d+1}

M. Karapetyan*

Yerevan Physics Institute, Yerevan, 0036 Armenia

**e-mail: meliq.karapetyan@gmail.com*

Received November 15, 2019; revised January 15, 2020; accepted February 28, 2020

Abstract—We present the calculation of the solution of recurrence relations for radial pullback of cubic interaction in AdS_{d+1} by using Wolfram Mathematica as an effective tool for computing large expressions. Investigation of the results of these calculations allows us to guess the ansatz for solution and then obtain it completely.

DOI: 10.1134/S1547477120050209

1. THE RECURRENCE RELATION

When considering a possible radial pullback scheme for the main object of cubic interaction in [1] we obtain the following objects defined recursively:

$$V^{p+1}(i_{p+1}) = \sum_{i_p=0}^{i_{p+1}} \varphi_{i_p} V^p(i_p),$$
$$V^{p+1}(i_{p+1}) = \sum_{k=0}^{\left[\frac{p}{2}\right]} \xi_k^{p+1}(i_{p+1})(a^2)^k (a^u)^{p-2k}, \quad (1)$$

where

$$\varphi_{i_k} = a^u[2(i_k + k) - s] + \left[a^2 - (a^u)^2\right]\partial_{a^u}. \quad (2)$$

Using (1), (2) we see that $\xi_k^{p+1}(i)$ satisfy the following recurrence relation

$$\xi_k^{p+1}(i) - \xi_k^{p+1}(i-1) = (2i + p + 1 + 2k - s)\xi_k^p(i) + (p + 1 - 2k)\xi_{k-1}^p(i). \quad (3)$$

In the following sections we will demonstrate steps for solving this equation.

2. MODELLING RECURRENCE RELATION USING WOLFRAM MATHEMATICA

To solve the (3), the (1) is modeled using Wolfram Matematica by creating function that will effectively compute (1) terms.

```
In[1]:= a0 = Superscript[a, 0];

In[2]:= psi[ps_, {i_, k_Integer}] := a0 * (2 * (i + k) - s) * ps + (a^2 - a0^2) D[ps, a0]
```

In the first step we model (2) then define several auxiliary functions for later use in the calculations. They are mainly for working with Wolfram Mathematica list objects such as slicing them or generating substitution indexes which will be used in summations. And finally in the last step we define (1) using objects from the previous steps.

```
In[20]:= slice[l_List, k_Integer] := l[[Table[n + k, {n, Length[l] - k}]]];
         inds[j_Integer] := Table[Subscript[i, k], {k, j}];
         inds[j_Integer, q_Integer] := slice[inds[q], j - 1];
         indsSub[j_Integer] := Table[{Subscript[i, k], k}, {k, j}]
         indsSub[j_Integer, q_Integer] := slice[indsSub[q], j - 1]
         summInds[len_Integer, bound_] := Module[{inds},
           Subscript[i, len + 1] = bound;
           inds = Table[{Subscript[i, k], 0, Subscript[i, k + 1]}, {k, 1, len}];
           Subscript[i, len + 1] =.;
           inds
          ]
```

```
In[46]:= psiProd[1] = psi[1, indsSub[1][[1]]];
         psiProd[end_Integer] := FoldList[psi, psiProd[1], indsSub[2, end]][[end]];
         summPsiProd[psiCount_Integer] :=
           FoldList[Sum, psiProd[psiCount],
             summInds[psiCount, Subscript[i, psiCount + 1]]][[psiCount + 1]];
```

Here $summPsiProd(p)$ function takes as argument p and computes the expression for $V^{p+1}(i_{p+1})$ using

$$V^{p+1}(i_{p+1}) = \sum_{i_p=0}^{i_{p+1}} \varphi_{i_p} V^{p}(i_p). \tag{4}$$

For better representability of the obtained results, a new function is introduced which will take as argument an expression and return equivalent expression with all coefficients of given variables factorized.

```
In[ ]:= factorPolynom[polynom_, vars_List] :=
         Module[{coefList, dims, powsInds, generator, powIter, i, varMatrix, polyMatrix},
          (*factorize coefficients of polynom*)
          coefList = CoefficientList[polynom, vars] // Factor;
          (*generate variable matrix for polynom*)
          dims = Dimensions[coefList];
          powsInds = Table[Subscript[i, n], {n, Length[vars]}];
          generator = Times @@ (vars^powsInds);
          powIter[{k_, b_}] := {k, 0, b};
          (* This line generates all possible combinations of vars in given dims *)
          varMatrix = Table @@ Prepend[
              Map[powIter, Transpose[{powsInds, dims - 1}]],
              generator];

          polyMatrix = varMatrix * coefList;
          (* returns the resulting polynom with factorized coefficients*)
          Total[polyMatrix, Length[vars]]
         ]
```

The final computing function for (1) is

The $V[p]$ function above computes $V^{p+1}(i_{p+1})$. Now we can effectively compute the values by simply plugging values for different p

```
In[60]:=
         V[p_] :=
          factorPolynom[#, {a, a0}] & @ Collect[summPsiProd[p - 1], {a0, a}, Simplify] //
            Evaluate // HoldForm
```

3. COMPUTATION OF SUMMATIONS AND ANSATZ FOR $\xi_k^{p+1}(i)$

Using Mathematica function for (4) we compute it for initial values of $p = 1, 2, 3, 4, 5, 6, 7, 8\ldots$

$$V^2(i_2) = \sum_{i_1=0}^{i_2} \varphi_{i_1} \mid 0 >= (1+i_2)(2-s+i_2)a^u \mid 0 >, \tag{5}$$

$$\begin{aligned} V^3(i_3) = \sum_{i_1=0}^{i_2} \varphi_{i_2} V^2(i_2) = \frac{1}{6}a^2(1+i_3)(2+i_3)(6-3s+2i_3) \mid 0 >, \\ +\frac{1}{2}(1+i_3)(2+i_3)(2-s+i_3)(3-s+i_3)\left(a^u\right)^2 \mid 0 >, \end{aligned} \tag{6}$$

$$V^4(i_4) = \sum_{i_3=0}^{i_4} \varphi_{i_3} V^3(i_3) = \frac{1}{6}a^2(1+i_4)(2+i_4)(3+i_4)(4-s+i_4)(6-3s+2i_4)a^u \mid 0 > . \tag{7}$$

Investigating these we discover the following non-trivial ansatz for $\xi_k^{p+1}(i)$

$$\xi_k^{p+1}(i) = \frac{1}{(p-2k)!}(i+1)_p(2k+2+i-s)_{p-2k}P_k(i), \quad (8)$$

where $P_k(i) \sim i^k + \ldots$ is now *p- independent polynomial of order k* and we introduced Pochhammer symbols

$$(a)_n = \frac{\Gamma(a+n)}{\Gamma(a)} = a(a+1)\ldots(a+n-1). \quad (9)$$

Inserting (8) in Eq. (3) we obtain equation for $P_k(i)$:

$$(i+2k)P_k(i) - iP_k(i-1) = (i+2k-s)P_{k-1}(i). \quad (10)$$

4. SOLVING RECURRENCE RELATION FOR P_k

To solve (10) we introduce a more convenient normalization of our polynomials with additional $2k$ order factor:

$$\mathcal{P}_k(i) \equiv (i+1)_{2k}P_k(i), \quad (11)$$

we arrive to the following simple equation with boundary condition:

$$\mathcal{P}_k(i) - \mathcal{P}_k(i-1) = (i+2k-1)(i+2k-s)\mathcal{P}_{k-1}(i), \quad (12)$$

$$\mathcal{P}_0(i) = P_0(i) = 1. \quad (13)$$

The solution of this equation can be written in the form of multiple sums:

$$\mathcal{P}_k(i) = \sum_{i\geq i_k\geq i_{k-1}\geq i_{k-2}\ldots\geq i_1\geq 0}\prod_{n=1}^{k}(i_n+2n-1)(i_n+2n-s), \quad (14)$$

or as a solution for differential equation for generating function

$$\mathcal{P}_k(y) \equiv \sum_{i=0}^{\infty}\mathcal{P}_k(i)y^i, \quad (15)$$

where we introduced formal variable y with $|y| < 1$ for production of the boundary condition:

$$\mathcal{P}_0(y) = \sum_{i=0}^{\infty}y^i = \frac{1}{1-y}. \quad (16)$$

For this generation function, we obtain from recurrence relation (12) the equation

$$(1-y)\mathcal{P}_k(y) = \left(y\frac{d}{dy} + 2k - 1\right) \times \left(y\frac{d}{dy} + 2k - s\right)\mathcal{P}_{k-1}(y). \quad (17)$$

Solving recursively and using (16) we can write the solution in the form:

$$\mathcal{P}_k(y) = y^{-(2k+1)}\left[\frac{y^4}{1-y}\frac{d}{dy}y^s\frac{d}{dy}y^{-s}\right]^k\frac{y^2}{1-y}. \quad (18)$$

Finally, we can write (8) in term of $\mathcal{P}_k(i)$

$$\xi_k^{p+1}(i) = \frac{1}{(p-2k)!}(2k+i+1)_{p-2k} \times (2k+2+i-s)_{p-2k}\mathcal{P}_k(i). \quad (19)$$

5. CONCLUSIONS

We have successfully solved the recurrense relation which was found during the radial pullback of cubic interaction in AdS_{d+1} and presented the modelling procedure using wolfram mathematica to compute large expressions and display them in a beautifully structured manner, which makes the non trivial ansatz visible.

REFERENCES

1. M. Karapetyan, R. Manvelyan, and R. Poghossian, Nucl. Phys. B **950**, 114876 (2020); arXiv: 1908.07901
2. 2. R. Manvelyan, R. Mkrtchyan, and W. Rühl, Nucl. Phys. B **872**, 265 (2013); arXiv: 1210.7227