South Tangerang, April 16, 2025

Dear Editor of Computer Physics Communications (CPC),

Thank you for sending us the second round of referee reports for our article, "pyBoLaNO: A Python symbolic package for normal ordering involving bosonic ladder operators" (COMPHY-D-25-00012). We are delighted that our work may be accepted pending appropriate revisions in response to Reviewer #1's comments. We also thank Reviewer #2 for their final remark.

We address the remaining comments from Reviewer #1 by modifying some parts of the manuscript and updating our GitHub repository. The revised texts in the manuscript are indicated in red in the compiled PDF file. In the LaTeX source file, we use a self-defined environment `\begin{revision2}...\end{revision2}`. The revision indicator can be safely removed by redefining this environment in the preamble (as we have for the first `revision` environment). The detailed answers to each comment/question, alongside the corresponding changes made to the manuscript, are enclosed with this response letter.

Thank you once again for considering our work for publication in CPC.

Corresponding author:

Hendry Minfui Lim

hendry01@ui.ac.id

Research Center for Quantum Physics, National Research and Innovation Agency (BRIN), South Tangerang 15314, Indonesia

Department of Physics, Faculty of Mathematics and Natural Sciences, Universitas Indonesia, Depok 16424, Indonesia

REPLY TO REPORT OF REVIEWER 1

REVIEWER'S REMARK:

This paper describes an efficient algorithm to bring strings of bosonic operators in normal order. It has similar capabilities as the sympy.physics package but is more efficient.

AUTHORS:

Thank you for your appreciation to this work and for the precious time you spent in reviewing the manuscript and codes.

QUESTION/COMMENT 1:

It would be nice if the package could be combined with symbolic use of normal modes (q_i, q_j) and sums over i, j etc., rather than explicit terms q_1, q_2. That does not seem to be the case at the moment.

REPLY:

Thank you for your comment and suggestion. As we have stated in **Section 3.2** of the previous version of the manuscript, the subscript k in $\hat{b}_k$ is treated as a `Symbol` in `SymPy`. Therefore, users may utilize *any* symbol as the subscript, including the dummy index of a sum or product.

Arguably, the most computationally efficient way to normal-order a sum or product written in the sigma or pi notation is to **first normal-order the summand expression, then carry out the sum or product**. As a simple example, the normal ordering

$$\mathcal{N}\left(\sum_{j=a}^{b}\sum_{k=c}^{d}\hat{b}_j\hat{b}_k\hat{b}_j^\dagger\hat{b}_k^\dagger\right)=\sum_{j=a}^{b}\sum_{k=c}^{d}\left(1+\hat{b}_j^\dagger\hat{b}_j\right)\left(1+\hat{b}_k^\dagger\hat{b}_k\right) \tag{1}$$

can be reproduced in `SymPy` with the help of our package as follows.

```
import pybolano as bl
import sympy as sm

j, k, a, b, c, d = sm.symbols("j k a b c d")

b_j, bd_j = bl.ops(j)
b_k, bd_k = bl.ops(k)

summand = b_j*b_k*bd_j*bd_k
```

```
summand_NO = bl.NO(summand)

sum = sm.Sum(summand_NO, (j,a,b), (k,c,d))

print(sm.latex(sum))

Out: \sum_{\substack{a \leq j \leq b\\c \leq k \leq d}} \left(1 +
↪  {b^{\dagger}_{j}} {b_{j}} + {b^{\dagger}_{j}} {b^{\dagger}_{k}}
↪  {b_{j}} {b_{k}} + {b^{\dagger}_{k}} {b_{k}}\right)
```

The render of the LaTeX output is

$$\text{Out:} \qquad \sum_{\substack{a\leq j\leq b\\ c\leq k\leq d}} \left(1 + b_j^\dagger b_j + b_j^\dagger b_k^\dagger b_j b_k + b_k^\dagger b_k\right) \tag{2}$$

which is equivalent to the RHS of Eq. (1).

We recognize the lack of emphasis on this capability in the previous version of the manuscript and have added to **Section 4.1** (of the revised manuscript) the demonstration of using symbols as subscripts.

We have also noted how our package's ladder operator base object, `pybolanoOp`, is compatible with other `SymPy` expressions, such as the sigma and pi notations, which are ubiquitous in many-body cases. For example, with the variables in the above code block, we have

```
print(sm.latex(sm.Sum(b_j, (j,1,3)).doit())

Out: {b_{1}} + {b_{2}} + {b_{3}}
```

or

$$\sum_{j=1}^{3} \hat{b}_j = \hat{b}_1 + \hat{b}_2 + \hat{b}_3 \tag{3}$$

which can be correctly produced with `sympy.Sum` alongside our ladder operator objects. This remark has been added to **Section 3.2**. We have also updated the tutorial file in the GitHub repository with commit `2f1aa49`.

BEFORE REVISION:

Section 3.2 of the first revision

... This function preprocesses `k` into `Symbol` to be used as the subscript of ladder operators.

AFTER REVISION:

Section 3.2 of the second revision

... This function preprocesses `k` into `Symbol` to be used as the subscript of ladder operators.

Since the objects inherit from `sympy.Expr` without any modifications to methods handling interactions between `SymPy` objects, they are compatible with other expression objects. In particular, they are compatible with the sigma and pi notations, implemented as `sympy.Sum` and `sympy.Product`, respectively. That is, the summation and product can be explicitly carried out, e.g. $\sum_{k=1}^{3} \hat{b}_k = \hat{b}_1 + \hat{b}_2 + \hat{b}_3$ and $\prod_{k=1}^{3} \hat{b}_k = \hat{b}_1 \hat{b}_2 \hat{b}_3$.

QUESTION/COMMENT 2:

The output is rendered in LaTeX, and this is how much of the results in the document are illustrated. I think it would be nice if those results would then be typeset such that it is easier to read.

REPLY:

Thank you for pointing out the LaTeX issue. We have added the LaTeX renders of the outputs (and inputs, where necessary) after the corresponding code blocks in the manuscript. These revisions are not all specified in this response due to their large number. The following illustrates **one example** of the revised output representations.

BEFORE REVISION:

Section 4.1 of the first revision

```
print(latex(
bl.normal_ordering(b * bd * b)
))

Out: b_{} + {b^\dagger_{}} b_{}^{2}
```

AFTER REVISION:

Section 4.1 of the second revision

```
print(latex(
bl.normal_ordering(b * bd * b)
))

Out: b_{} + {b^\dagger_{}} b_{}^{2}
```

`Input render:`

$$\hat{b}\hat{b}^{\dagger}\hat{b}$$

`Output render:`

$$b + b^{\dagger}b^{2}$$

REPLY TO REPORT OF REVIEWER 2

REVIEWER'S REMARK:

I am satisfied with the authors replies.

AUTHORS:

Thank you for your appreciation to this work and for the precious time you spent in reviewing the manuscript and codes.