A Newman-Penrose Calculator for Instanton Metrics

T. Birkandan*
Istanbul Technical University,
Faculty of Science and Letters,
Dept. of Physics, Maslak 34469, Istanbul, Turkey

Abstract

We present a Maple11+GRTensorII based symbolic calculator for instanton metrics using Newman-Penrose formalism. Gravitational instantons are exact solutions of Einstein's vacuum field equations with Euclidean signature. The Newman-Penrose formalism, which supplies a toolbox for studying the exact solutions of Einstein's field equations, was adopted to the instanton case and our code translates it for the computational use.

Keywords : Gravitational instantons; Newman-Penrose formalism; Symbolic computation

PACS: 04., 04.62.+v

1 Introduction

The interest in symbolic computational study of general relativity is growing rapidly as the capacity of the the computer systems increase. The computer is no longer an apparatus for the numerical relativist, but as the symbolic manipulators get easier to use, the more researchers get into the subject by using these systems. Comprehensive historical reviews can be found in references [1, 2, 3]. Another important point to consider is the specialized packages which run under these platforms. GRTensorII [4], being a widespread package in general relativity led significant progress in the area [5, 6, 7, 8, 9, 10].

The Newman-Penrose (NP) formalism supplies a toolbox for investigating the exact solutions of the Einstein's field equations [11, 12]. Goldblatt has developed NP formalism for gravitational instantons based on the $SU(2) \times SU(2)$ spin structure of positive definite metrics [13, 14]. In the gravitational instanton case, the gravitational field decomposes into its self-dual and anti-self-dual parts and this decomposition is natural in the spinor approach which necessitates two independent spin frames for the spinor structure of 4-dimensional Rieman manifolds with Euclidean signature [15].

Gravitational instantons are exact solutions of Einstein's vacuum field equations with Euclidean signature. They are analogous to the Yang-Mills instantons, the finite action solutions of the classical Yang-Mills equations. They admit hyper-Kähler structure [15, 16]. For a detailed overview one can see Eguchi et al's review and the articles cited in this paper [17].

Aliev and Nutku applied differential forms to the NP formalism for gravitational instantons which made the formalism more suitable for the symbolic computation [15].

*E-mail : birkandant@itu.edu.tr

2 The program

Our program, NPInstanton is designed for calculating physical and mathematical quantities for instanton metrics using Newman-Penrose formalism. It is coded under Maple 11 and GRTensorII package. The quantities which can be calculated are:

- massless scalar equation,

- massless Dirac equation,

- source-free Maxwell equations,

- covariant and contravariant Dirac γ matrices,

- coframe $l(= l_\mu dx^\mu)$ and $m(= m_\mu dx^\mu)$,

- Ricci rotation coefficients,

- Weyl scalars,

- trace-free Ricci scalars,

- spinor equivalent of the connection 1-forms,

- basis 2-forms,

- curvature 2-forms,

- integrands of the Euler number and the Hirzebruch signature curvature part integrals,

- Petrov class of the space.

As one can see, some of the objects could be calculated by standard means and without using a signature-dependent package. But for the sake of completeness, we added these features to the program. By these, the program becomes a complete symbolic calculator for an instanton metric. The NP calculator of GRTensorII is not designed for the Euclidean signature and could give unexpected results. Therefore, a complete NP based calculator, combining the power of GRTensorII and NP formalism for these special metrics is useful.

The program uses,

1. GRTensorII Package: Several objects (Ricci scalar, covariant Weyl tensor, etc.) are calculated by this package in our program. No specific knowledge of this package is needed for using NPInstanton. Our program creates the metric file needed for the calculation itself using the NP legs given in the input file and writes it to the metric directory of GRTensorII as "npinstanton.mpl". This metric file then can be used in GRTensorII independently.

2. DifferentialGeometry Package of Maple 11: The definition of the wedge product is supplied by this package in our program. The wedge product definition is also supplied by "difforms" package in Maple but it has conflicts with linear algebraic quantities. This package is excluded in a limited version of the program for the users of older Maple versions.

3. linalg package of Maple: This rather old internal package is used for an eigenvalue calculation in Petrov classification section.

The program is set for a computer having a 512 Mb of RAM (Average value for today's personal computers). If the system has less memory, the user must change the line

```
kernelopts(gcfreq=10^7):
```

of the **npinstanton.mws** file to a lower value (10^6 is the standard value of Maple). For a computer having

larger than 1 Gb of memory, the user may change the gcfreq value as 10^8. "gc" is the abbreviation of "garbage collection" and it is the Maple's internal routine which cleans the memory after an amount of memory is allocated. For a computer having a large amount of memory, one can increase the frequency of this process. The larger the gc frequency value results in more memory to be wasted but for a system having a large amount of memory it increases the performance for some calculations.

The programming style is procedure-based. Each command is a Maple procedure which can call other calculation procedures and inherit their outputs. For example, when the **dirac()** command is given for the calculation of massless Dirac equation, the program calls **dirac()** procedure. The calculation of Dirac equations needs the information on γ matrices. Therefore, **dirac()** procedure calls **gammamatricespless()** procedure for this calculation and its output is sent to **dirac()** procedure. The suffix "**-pless**" (printless) means that this procedure is a "secondary" one and it does not print its output to the screen. When **dirac()** procedure finishes execution, it removes the information that was inherited from **gammamatricespless()** procedure and sends the user only its permitted output as **dirac** vector.

The "secondary" procedures do the massive works of the main procedures and the main procedure generally contains the key directives. Therefore, this usage makes the program easier to track. In addition to this, if a user wants to add another command to the program, a procedure of that command can be coded using these "secondary" procedures, or the user can add his/her own "secondary" procedures, as well.

Throughout the program, only the output variables of the procedures are the global ones and only these variables can be reached by the user at the end of the execution of the command. Therefore, the user should be aware of the output names of the procedures that are being used. This global variables are shown at the end of the execution of each command. For example, the output variable of **scalaroperator()** command will be shown as **scalarop** and it can be called in the session when needed.

The outputs of the auxiliary calculations are cleared from the memory by setting them as local variables, as these auxiliary calculations may occupy too much memory. For example, γ matrices information is not necessary if the user needs to know only the Dirac equation. Therefore, **dirac()** procedure clears the information about γ matrices after finishing the execution. If the user needs to know the γ matrices, **gammamatrices()** procedure is run by the user. As another example, in **conn1form()** procedure, only output values are connection 1-forms and the outputs from **coframepless()** and **riccirotcoeffpless()** procedures, as well as internal calculations are dismissed to gain memory for further calculations.

A few commands for simplification are added to the code which can be evaluated easily by an average personal computer (i.e. having a 512 Mb of memory), but it is always more convenient for the user to choose the right simplification technique for the problem after the calculation. The output must be regarded as a "raw material" for a simplification routine that is to be chosen by the user. The user having a computer with insufficient memory can extract the simplification routines from the program, simply by modifying it in an editor but it is not recommended as it may lead to miscalculation of some properties such as Petrov type or (anti-)self-duality.

One can reach the program files using the web site given in the "Final Remarks" section. Linux and Windows versions are available and for those who have older Maple versions, a limited version of NPInstanton which does not contain the **basis2form()**, **curv2form()** and **topologicalnumbers()** parts (parts that use DifferentialGeometry Package) is also supplied. The details on how to run the program is given in the README file which comes with the program files.

2.1 The input file

The input file contains the necessary definitions for the space and wave equations. General definitions (constants, etc.) can also be given in this file. The definitions should be given using Maple's syntax. The input file of our program plays the role of the metric file of GRTensorII.

The user first sets the coordinate names in the first section as,

```
firstcoordinate:= x:
```

```
secondcoordinate:= theta:
```

etc. as given in example input file. Then the components of the covariant NP legs are entered. **l_covar[1]** being the first component of the covariant l_μ leg and **l_bar_covar[1]** being the complex conjugate of **l_covar[1]**, etc.. Maple does not do the complex simplifications because they need assumptions that may cause wrong calculations. Therefore, the most appropriate way to define the legs is to give both by hand. The choice of NP legs is not unique for a metric. For the Eguchi-Hanson NP legs we can have[17]:

$$l = \frac{1}{\sqrt{2}}\left[\frac{1}{\sqrt{1-\frac{a^4}{r^4}}}dr - \frac{ir}{2}\sqrt{1-\frac{a^4}{r^4}}(d\xi + \cos\theta d\phi)\right], \tag{1}$$

$$m = \frac{re^{-i\xi}}{2\sqrt{2}}\left(d\theta + i\sin\theta d\phi\right). \tag{2}$$

Then the input forms are

```
l_covar[1]:=1/(sqrt(2)*A):
l_bar_covar[1]:=1/(sqrt(2)*A):
```

etc. as given in example input file. The A value being $\sqrt{1-\frac{a^4}{r^4}}$ was given in the general definitions section.

Spinor components can be given if the user will be calculating the Dirac equation. They can be chosen as [18],

$$\psi_1 = e^{i(m+\frac{1}{2})\xi}\Psi_1(r,\theta,\phi), \tag{3}$$

$$\psi_2 = e^{i(m+\frac{1}{2})\xi}\Psi_2(r,\theta,\phi), \tag{4}$$

$$\psi_3 = e^{i(m-\frac{1}{2})\xi}\Psi_3(r,\theta,\phi), \tag{5}$$

$$\psi_4 = e^{i(m-\frac{1}{2})\xi}\Psi_4(r,\theta,\phi) \tag{6}$$

and the input form is

```
spinorcomponent1:=exp(I*(m+(1/2))*xi)*psi1[r,theta,phi]:
```

etc. as given in example input file.

The scalar function can be given for the calculation of the scalar equation. This definition can be skipped if the user does not need to calculate this object. It can be chosen as

$$\varphi = e^{im\xi}\Phi(r,\theta,\phi) \tag{7}$$

for the Eguchi-Hanson space, φ being the scalar function and it is set as

```
scalarfunction:=exp(I*m*xi)*Phi(r,theta,phi):
```

in the input file.

2.2 Command definitions

The list of commands can be given as,

```
scalaroperator()
dirac()
maxwell()
gammamatrices()
coframe()
spinrotcoeff()
weylscalar()
tfricciscalar()
conn1form()
basis2form()
curv2form()
topologicalnumbers()
```

and the definitions of these commands are the following,

- scalaroperator():

This command calculates the massless scalar equation, finding the scalar operator. The scalar operator is given as

$$H\varphi \equiv \frac{1}{\sqrt{|g|}}\partial_\nu \sqrt{|g|} g^{\mu\nu}\partial_\mu \varphi. \tag{8}$$

Here, g is the determinant of the metric.

The procedure takes the scalar function (name: **scalarfunction**) from the input file.

The output to be used thereafter:

```
> scalarop;
```

For the massive case, one can equate this object to $M^2\varphi^2$, M being the mass of the scalar particle. As an additional property, this procedure is not dependent on the metric signature and it does not use the NP objects so it can be used for any space in four dimensions by extracting it from the program.

- dirac():

This command calculates the massless Dirac equations,

$$\gamma^\mu \nabla_\mu \psi = 0 \tag{9}$$

where

$$\nabla_\mu = \partial_\mu - \Gamma_\mu \tag{10}$$

and Γ_μ are the spin connections. The procedure takes the spinor vector components from the input file under these names: **spinorcomponent1** (ψ_1), **spinorcomponent2** (ψ_2), **spinorcomponent3** (ψ_3) and **spinorcomponent4** (ψ_4). The output to be used thereafter is the components of the "**dirac**" vector as (i=1, 2, 3, 4):

```
> dirac[i];
```

For the massive case, one can equate these objects to $\frac{M}{i}\psi$ vector, M being the spinor mass and $i \equiv \sqrt{-1}$.

- maxwell():

This command calculates the source-free Maxwell equations using the equations 102-109 of [15]. F_{ij}'s in the output are the usual Maxwell field matrix components. The output is set to be "**maxwell**" vector whose components are the source-free Maxwell equations as (i=1, 2, 3, 4):

```
> maxwell[i];
```

- gammamatrices():

This command finds the covariant and contravariant Dirac γ matrices and tests them with anticommutation relations

$$\{\gamma^{\mu}, \gamma^{\nu}\} = 2g^{\mu\nu}. \tag{11}$$

The output is "**gamma_contr**" (contravariant γ matrices: γ^{μ}) and "**gamma_covar**" (covariant γ matrices: γ_{μ}) vectors as (i=1, 2, 3, 4):

```
> gamma_contr[i];
> gamma_covar[i];
```

The calculation of the Dirac γ matrices could be put into the **dirac()** procedure but for further calculations involving higher dimensions one may need only these matrices. To find the γ matrix for the extra dimension, one can use these results and look for the γ matrix of the extra dimension using the anticommutation relations as in reference [19]. Therefore the procedure is a separate one.

- coframe():

This command calculates the coframe l $\equiv$**L** ($= l_{\mu}dx^{\mu}$) and m $\equiv$**M** ($= m_{\mu}dx^{\mu}$). The output variables are the following, "**_bar**" denoting the complex conjugate:

```
> L;
> M;
> L_bar;
> M_bar;
```

- riccirotcoeff():

This procedure calculates the Ricci rotation coefficients using equation 20 of [15]. The output can be used later by calling:

```
> npkappa; > nptau; > npsigma;
> nprho; > nppi; > npnu;
> npmu; > nplambda; > npgamma;
> npepsilon; > npalpha; > npbeta;
```

in the meaning of the Ricci rotation coefficients κ, τ, σ, ρ, π, ν, μ, λ, γ, ε, α, β respectively.

- weylscalar():

This command calculates the Weyl scalars using equation 68 of [15]. The output variables are:

```
> weylscalar0; > weylscalar1; > weylscalar2, > weylscalar3;
> weylscalar4; > weylscalartilde0; >weylscalartilde1;
> weylscalartilde2; > weylscalartilde3; > weylscalartilde4;
```

for Ψ_0, Ψ_1, Ψ_2, Ψ_3, Ψ_4, $\tilde{\Psi}_0$, $\tilde{\Psi}_1$, $\tilde{\Psi}_2$, $\tilde{\Psi}_3$, $\tilde{\Psi}_4$ respectively. "tilde" means the variable has a tilde in the meaning of a different spin frame.

Another property of this procedure is that it finds out the Petrov class of the space according to equations 120-122 of [15]. Petrov classification of Euclidean spaces were first studied by Hacyan [21] and then by Karlhede [22].

- tfricciscalar():

This command calculates the trace-free Ricci scalars using equation 69 of [15] and sets them to variables for a later use:

```
> tfricciscalar00; > tfricciscalar01; > tfricciscalar02;
> tfricciscalar10; > tfricciscalar11; > tfricciscalar12;
> tfricciscalar20; > tfricciscalar21; > tfricciscalar22;
> scalofcurv;
```

for Φ_{00}, Φ_{01}, Φ_{02}, Φ_{10}, Φ_{11}, Φ_{12}, Φ_{20}, Φ_{21}, Φ_{22}, scalar of curvature Λ respectively.

- conn1form():

This procedure calculates the spinor equivalent of the connection 1-forms given by equations 36-37 of [15]. The output can be reached by calling

```
> GAMMA00; > GAMMA01; > GAMMA10; > GAMMA11;
> GAMMAtilde0pr0pr; > GAMMAtilde0pr1pr;
> GAMMAtilde1pr0pr; > GAMMAtilde1pr1pr;
```

for Γ^0_0, Γ^1_0, Γ^0_1, Γ^1_1, $\tilde{\Gamma}^{0'}_{0'}$, $\tilde{\Gamma}^{1'}_{0'}$, $\tilde{\Gamma}^{0'}_{1'}$, $\tilde{\Gamma}^{1'}_{1'}$ respectively where "tilde" means the variable has a tilde and "**pr**" means "prime" in the meaning of a different spin frame.

Another property of the procedure is that, it finds out the gauge by checking the necessary and sufficient conditions for (anti-)self-duality namely, $\Gamma_{ab} \equiv 0$ implies self duality and $\tilde{\Gamma}_{x'y'} \equiv 0$ implies anti-self-duality.

- basis2form():

This command finds the basis 2-forms using the definitions in equation 49 of [15]. The output can be reached by

```
> L00; > L01; > L10;
> Ltilde0pr0pr; > Ltilde0pr1pr; > Ltilde1pr0pr;
```

for L_0^0, L_0^1, L_1^0, $\tilde{L}_{0'}^{0'}$, $\tilde{L}_{0'}^{1'}$, $\tilde{L}_{1'}^{0'}$ respectively. Here, "tilde" means the variable has a tilde and "**pr**" means "prime" in the meaning of a different spin frame.

- curv2form():

This command calculates the curvature 2-forms using equations 90-91 of [15] and sets them to these variables:

```
> Theta00; > Theta01; > Theta10;
> Thetatilde0pr0pr; > Thetatilde0pr1pr; > Thetatilde1pr0pr;
```

for Θ_0^0, Θ_0^1, Θ_1^0, $\tilde{\Theta}_{0'}^{0'}$, $\tilde{\Theta}_{0'}^{1'}$, $\tilde{\Theta}_{1'}^{0'}$ respectively where "tilde" means the variable has a tilde and "**pr**" means "prime" in the meaning of a different spin frame.

- topologicalnumbers():

This command calculates the integrands of the Euler number and the Hirzebruch signature curvature part integrals using the relations 115-117 of [15] namely,

$$\begin{aligned}\chi &= \frac{1}{4\pi^2}\int_{\mathcal{M}}\Big[\,|\,\Psi_0\,|^2+4\,|\,\Psi_1\,|^2+3\,\Psi_2^2+|\,\tilde{\Psi}_0\,|^2 \\ &\quad +4\,|\,\tilde{\Psi}_1\,|^2+3\,\tilde{\Psi}_2^2-2\,(\,|\,\Phi_{00}\,|^2+|\,\Phi_{02}\,|^2\,) \\ &\quad -4\,(\,|\,\Phi_{01}\,|^2+|\,\Phi_{11}\,|^2+|\,\Phi_{12}\,|^2-3\Lambda^2\,)\Big]\,l\wedge\bar{l}\wedge m\wedge\bar{m} \qquad (12)\end{aligned}$$

$$\begin{aligned}\tau &= -\frac{1}{6\pi^2}\int_{\mathcal{M}}\Big[\,|\,\Psi_0\,|^2+4\,|\,\Psi_1\,|^2+3\,\Psi_2^2-|\,\tilde{\Psi}_0\,|^2 \\ &\quad -4\,|\,\tilde{\Psi}_1\,|^2-3\,\tilde{\Psi}_2^2\Big]\,l\wedge\bar{l}\wedge m\wedge\bar{m}\;-\eta_s(\partial\mathcal{M}) \qquad (13)\end{aligned}$$

$\eta_s(\partial\mathcal{M})$ being the eta-invariant and this value will not be taken into consideration in the program. The output can be reached by calling

```
> eulernumber_integrand;
> hirzebruch_signature_integrand;
```

These numbers have a special importance for the Atiyah-Patodi-Singer index theorem of operators on manifolds with boundary [23][24][25][26].

3 Examples

In this section, we will apply our program to two instanton metrics and calculate some objects using the special commands. Lengthy output values are not written to avoid distracting the reader's attention.

3.1 Example 1: Calculations for the Eguchi-Hanson metric

Eguchi-Hanson instanton [17] is the most similar to the Yang-Mills instanton of Belavin et al. [27] and the metric is given as

$$ds^2 = \frac{1}{1-\frac{a^4}{r^4}}dr^2 + r^2(\sigma_x^2+\sigma_y^2) + r^2(1-\frac{a^4}{r^4})\sigma_z^2. \tag{14}$$

Here,

$$\begin{aligned}
\sigma_x &= \frac{1}{2}(-\cos\xi d\theta - \sin\theta\sin\xi d\phi), \\
\sigma_y &= \frac{1}{2}(\sin\xi d\theta - \sin\theta\cos\xi d\phi), \\
\sigma_z &= \frac{1}{2}(-d\xi - \cos\theta d\phi).
\end{aligned} \tag{15}$$

and the dyad was given in eq. 1 and eq. 2. We run our program in Maple and type the name of the input file: **eguchihanson**

Now, let us calculate the connection 1-forms:

```
npinstanton> conn1form();
```

The program calculates and shows the calculated values. Then the gauge is found by the program as:

```
SELF DUAL GAUGE because all connection 1-forms without
tilde are zero
```

before prompting for another calculation, the procedure shows the output variables which can be used for a later calculation. for example, let us call Γ_0^1 of equation 36 of [15]:

```
npinstanton> GAMMA01;
```

To find the Weyl scalars and the Petrov class, one can run,

```
npinstanton> weylscalar();
```

After the Weyl scalars are shown, the Petrov type is found to be:

```
Petrov-type D according to anti-self-dual part
```

All these results agree with the literature [22].

We can find the scalar operator by using

```
npinstanton> scalaroperator()
```

command. The scalar equation can be solved in terms of hypergeometric equations in four dimensional case by the command,

```
npinstanton> pdsolve(scalarop,INTEGRATE);
```

We can easily find the solution for the scalar equation with one extra dimension added trivially if this dimension is a Killing direction (i.e. $ds^2 = -dt^2 + ds_4^2$, ds_4^2 being the original instanton metric similar to the case in reference [19]). As an example, we just add $-k_t^2\Phi(r,\theta)$ to the equation after dividing by the exponential part which comes from the solution of the Killing directions:

```
npinstanton> pdsolve(scalarop/(exp((m*xi+n*phi)*I))
             -kt*kt*Phi(r,theta),INTEGRATE);
```

Here, k_t is the eigenvalue corresponding to the extra dimension. We see that the addition of the extra dimension results in a more singular equation, confluent Heun equation as the solution of the radial part. The occurrence of Heun equations in higher dimensional solutions is known in the literature (see [19] and references therein) and our example is a new one.

3.2 Example 2: Calculations for the Nutku helicoid metric

The NP legs of the Nutku helicoid metric can be chosen as [28],

$$l^\mu = \frac{a\sqrt{\sinh 2x}}{2}(1, i, 0, 0), \tag{16}$$

$$m^\mu = \frac{1}{\sqrt{\sinh 2x}}(0, 0, \cosh(x - i\theta), i\sinh(x - i\theta)). \tag{17}$$

This is an example of a multi-center metric. This metric reduces to the flat metric if we take $a = 0$. Since this solution has curvature singularities, it has not been studied extensively aside from four articles [18][19][20][29].

Let us calculate the basis 2-forms:

```
npinstanton> basis2form();
```

After the calculation, to call L_0^0 of equation 49 of [15],

```
npinstanton> L00;
```

and the output will be

-$\frac{1}{2}Ia^2\cosh(x)\sinh(x)dx1 \wedge dx2$-$\frac{1}{2}Idx3 \wedge dx4$

dx's were defined previously, at the very beginning of the session in terms of the real coordinates. We can calculate the Dirac equation by,

```
npinstanton> dirac();
```

and

```
npinstanton> dirac[3];
```

calls the third component of the Dirac equation vector which can be simplified and used for calculations.

We can calculate the connection 1-forms as,

```
npinstanton> conn1form();
```

For this definition the gauge turns out to be:

```
ANTI-SELF DUAL GAUGE because all connection 1-forms with
tilde are zero
```

and the connection 1-forms are shown.

For the Weyl scalars and the Petrov class, one can run,

```
npinstanton> weylscalar();
```

After the Weyl scalars are shown on the screen, the Petrov type is found to be:

```
Petrov-type I according to self-dual part
```

The whole results agree with the literature [28].

Now let us take a metric of the form $ds^2 = -dt^2 + ds_4^2$, ds_4^2 being the original Nutku helicoid metric. In this case, we need one more γ matrix for the extra dimension (γ^t). Firstly, let us use our command to find four dimensional γ matrices:

```
npinstanton> gammamatrices();
```

It is clear that the extra γ matrix will be diagonal as the four dimensional ones are non-diagonal. Then we can write these commands to form equations using $\{\gamma^\mu, \gamma^\nu\} = 2g^{\mu\nu}$:

```
extragamma:=Matrix([[ x1, x2,  0,  0 ],[ x3, x4,  0,  0 ],
                    [ 0,  0,  x5, x6 ],[ 0,  0,  x7, x8 ]]):
```

We have eight unknowns, so we will have eight equations:

```
eqn1:=(gamma_contr[1].extragamma+extragamma.gamma_contr[1])[1,3];
eqn2:=(gamma_contr[1].extragamma+extragamma.gamma_contr[1])[1,4];
eqn3:=(gamma_contr[1].extragamma+extragamma.gamma_contr[1])[2,3];
eqn4:=(gamma_contr[1].extragamma+extragamma.gamma_contr[1])[2,4];
eqn5:=(gamma_contr[2].extragamma+extragamma.gamma_contr[2])[1,3];
eqn6:=(gamma_contr[2].extragamma+extragamma.gamma_contr[2])[1,4];
eqn7:=(gamma_contr[2].extragamma+extragamma.gamma_contr[2])[2,3];
eqn8:=(gamma_contr[2].extragamma+extragamma.gamma_contr[2])[2,4];
```

and we solve them,

```
sol:=solve({eqn1=0,eqn2=0,eqn3=0,eqn4=0,
            eqn5=0,eqn6=0,eqn7=0,eqn8=0}
          ,{x1,x2,x3,x4,x5,x6,x7,x8});
```

Then, we have the form of the unknown matrix:

```
preresult:=subs(sol,extragamma);
```

To satisfy $\{\gamma^t, \gamma^t\} = -2$, we can set $x5 = i, x8 = -i$:

```
extramatrix:=subs(x5=I,x8=-I,preresult);
```

Then we can use this matrix with the name "**extramatrix**".

4 Final remarks

We introduced a Maple11+GRTensorII based symbolic calculator consisting of procedures for instanton metrics using a Euclidean Newman-Penrose formalism. A limited version which can run in older versions of Maple is also available.

As the program consists of procedures for different calculations, development of the program according to the user's needs is easy by adding procedures.

The program and sample files can be downloaded from the address:

```
https://github.com/tbirkandan/NPInstanton
```

Acknowledgement

The author would like to express his sincere to thanks to Prof. Mahmut Hortaçsu for help and support and Profs. Ayşe H. Bilge and Neşe Özdemir for scientific assistance and Fuat İ. Vardarlı for technical assistance throughout this work. This work is supported by TUBITAK, the Scientific and Technological Council of Turkey.

References

[1] I. Cohen, I. Frick, J. E. Åman, "10th International Conference on General Relativity and Gravitation", B. Bertotti et al. (eds.), 139 (1984)

[2] J. Grabmeier, E. Kaltofen, U. Weispfennig (eds.), "Handbook of Computer Algebra", Springer (2001)

[3] F.W. Hehl, R. Puntigam, H. Ruder (eds.), "Relativity and Scientific Computing: Computer Algebra, Numerics, Visualization", Springer-Verlag Telos (1996)

[4] P. Musgrave, D. Pollney and K. Lake, *Fields Institute Commun.* **15**,313 (1996). See also http://www.grtensor.org/

[5] N. Pelavas, N. Neary, K. Lake, *Class.Quant.Grav.* **18**, 1319 (2001)

[6] M. Ishak, L. Chamandy, N. Neary, K. Lake, *Phys. Rev. D* **64,** 024005 (2001)

[7] K. Santosuosso, D. Pollney, N. Pelavas, P. Musgrave, K. Lake, *Comp. Phys. Commun.* **115**, 381 (1998)

[8] M.S.R. Delgaty, K. Lake, *Comp. Phys. Commun.* **115**, 395 (1998)

[9] D. N. Vulcanov, *Comp. Phys. Commun***. 154**, 205 (2003)

[10] P. Musgrave, K. Lake, *Class. Quantum Grav.* **14**, 1285 (1997)
and see http://www.grtensor.org/papers.html for more.

[11] E. T. Newman, R. Penrose, *Jour. Math. Phys.* **3**, 566 (1962)

[12] E. T. Newman, R. Penrose, *Jour. Math. Phys.* **4**, 998 (1962)

[13] E. Goldblatt, *Gen. Rel. and Grav.* **26**, 979 (1994)

[14] E. Goldblatt, *Jour. Math. Phys.* **35**, 3029 (1994)

[15] A. N. Aliev, Y. Nutku, *Class. Quant. Grav.* **16**, 189 (1999)

[16] G. W. Gibbons, P. J. Ruback, *Commun. Math. Phys.* **115**, 267 (1988)

[17] T. Eguchi, P. B. Gilkey, A. Hanson, *Phys. Rep.* **66**, 213 (1980)

[18] Y. Sucu, N. Ünal, *Class. Quant. Grav.* **21**, 1443 (2004)

[19] T. Birkandan, M. Hortaçsu, *J. Phys. A: Math. Theor.* **40**, 1 (2007)

[20] T. Birkandan, M. Hortaçsu, *J. Math. Phys.* **48**, 092301 (2007)

[21] S. Hacyan, *Phys. Lett. A* **75**, 23 (1979)

[22] A. Karlhede, *Class. Quant. Grav.* **3**, L1 (1986)

[23] M. F. Atiyah, V. K. Patodi and I. M. Singer, *Math. Proc.Camb.Phil. Soc.* **77**, 43 (1975)

[24] M. F. Atiyah, V. K. Patodi and I. M. Singer, *Math. Proc. Camb.Phil. Soc.***77**, 405 (1975)

[25] M. Hortaçsu, K.D. Rothe,B. Schroer, 1*Nucl. Phys. B***17**, 530 (1980)

[26] M. Hortaçsu, *Lettere al Nuovo Cim.* **36**, 109 (1983)

[27] A. A. Belavin, A. M. Polyakov, A. S. Schwarz, Yu. S. Tyupkin, *Phys. Lett. B***59**, 85 (1975)

[28] A. N. Aliev, M. Hortaçsu, J. Kalaycı, Y. Nutku, *Class. and Quant. Grav.* **16**, 631 (1999)

[29] V. M. Villalba, *Jour. of Phys.:Conference Series* **24**, 136 (2005)