A short tutorial for using `TrackType:7` in `Bertini`

Jonathan Hauenstein

February 25, 2015

The main computation performed via `TrackType:7` in `Bertini` [1] is the following problem addressed in [2]: given witness sets for $A_1 \subset \mathbb{C}^{n_1}, \ldots, A_k \subset \mathbb{C}^{n_k}$ and a polynomial system $F : \mathbb{C}^{n_1+\cdots+n_k+M} \to \mathbb{C}^{\ell}$, compute

$$(A_1 \times \cdots \times A_k \times \mathbb{C}^M) \cap \mathcal{V}(F).$$

In order to perform this computation in `Bertini`, one first needs to computes witness sets for each A_i. For example, we use `TrackType:1` which performs a numerical irreducible decomposition here, but note that this could be also be computed via `TrackType:6` by using `ConstructWitnessSet:1`. Since `Bertini` writes over local files each time it is ran, it is best to compute witness sets for each A_i in separate directories.

For example, suppose $k = 2$ with $A_1 = \mathcal{V}(x_1 - x_2^2) \subset \mathbb{C}^2$, $A_2 = \mathcal{V}(y_1^2 + y_2^2 - 1) \subset \mathbb{C}^2$, and

$$F(x_1, x_2, y_1, y_2, z_1, z_2, z_3) = \begin{bmatrix} z_2^2 - x_1 z_2 + y_2 \\ z_3^2 - z_1 z_2 + x_2 - y_1 \end{bmatrix}.$$

In the files associated with this tutorial, we did the following for A_1:

(1) Created the folder `A1`

(2) In `A1`, created the file `input_A1` which is the following:

```
CONFIG
 TrackType:1;
END;
INPUT
 variable_group x1,x2;
 function f;
 f = x1 - x2^2;
END;
```

(3) Executed `Bertini` with the input file `input_A1`, e.g., $ `bertini input_A1`

We used a similar setup for A_2:

(1) Created the folder `A2`

(2) In `A2`, created the file `input_A2` which is the following:

```
CONFIG
 TrackType:1;
END;
INPUT
 variable_group y1,y2;
 function g;
 g = y1^2 + y2^2 - 1;
END;
```

(3) Executed `Bertini` with the input file `input_A2`, e.g., $ `bertini input_A2`

Now that we have witness sets for A_1 and A_2, we create the main `input` file needed to compute $(A_1 \times A_2 \times \mathbb{C}^3) \cap \mathcal{V}(F)$.

```
CONFIG
 TrackType:7;
END;
INPUT
 variable_group x1,x2,y1,y2,z1,z2,z3;
 function f; % for A1
 function g; % for A2
 function F1,F2; % new polynomials
 f = x1 - x2^2;
 g = y1^2 + y2^2 - 1;
 F1 = z2^2 - x1*z2 + y2;
 F2 = z3^2 - z1*z2 + x2 - y1;
END;
```

Note that we have concatenated the list of variables and declared the polynomials in order.

Upon running `Bertini` with this `input` file, the following is displayed to the screen:

```
*************** Regeneration Extension ***************

Please enter the number of nontrivial components (-1 to quit):
```

We enter `2` since $k = 2$ and then the following is displayed:

```
NOTE: Regeneration extension is only implemented for generically
reduced components (both input and output)!

NOTE: Regeneration extension assumes the witness sets for the
2 components are independent!

Setup for component 1 of 2.
Please enter the name of the corresponding input file or type
'quit' or 'exit' (max of 255 characters):
```

As stated in the first note, `Bertini` will only perform the computation when the components A_i are generically reduced with respect to the corresponding input system. This first note also states that the current implementation will only output generically reduced components.

The second note states that this implementation assumes that the witness sets are independent. In particular, this means that the slices are distinct. For example, say, you are performing computations with m components of the same polynomial system. It is best to solve using `Bertini` m times to generate m different witness sets, one for each of the components. This can be accomplished, for example, by performing the numerical irreducible decomposition `TrackType:1` m times. Another option is to first compute a numerical irreducible decomposition. Then, for each component, print the witness point set via `TrackType:4` and then reconstruct a witness set from the witness point set using `TrackType:6` with `ConstructWitnessSet:1`.

Now, returning to the menu, we enter the name of the first `input` file, namely `A1/input_A1`. After verifying the existence of that file, the following is then displayed:

```
Please enter the name of the corresponding witness_data file or type
'quit' or 'exit' (max of 255 characters):
```

With this setup, the name of this file is `A1/witness_data`. After reading this file, we then to select the dimension and component. The menu for selecting the dimension is:

```
*************** Components to Regenerate ***************

Dimension 1: 1 classified component
-----------------------------------------------------
   degree 2: 1 component


   Please select a dimension to regenerate (-1 to quit):
```

In this case, we enter `1` bringing up the following:

```
Dimension 1: 1 classified component
-----------------------------------------------------
   component 0 has degree 2 (gen. reduced: Yes)


   Please select a component to regenerate (-1 to quit, -2 to regenerate all):
```

Note that one may select a particular irreducible component or their union via the `-2` option.

This completes the selection for the first component and a similar process is repeated to select the second component. Once selected, `Bertini` performs some initial tests before performing the intersection. After computing the witness point superset, `Bertini` needs to "reclassify" the points with respect to the whole system and only keeps the ones which are generic points on generically reduced components, i.e., have multiplicity one. The user is reminded of twice, first by the witness set summary:

```
*********** Multiplicity 1 Witness Set Summary ************

NOTE: nonsingular vs singular is based on rank deficiency
and identical endpoints

|codim| witness points | nonsingular | singular
-------------------------------------------------
| 4   |   16           |  16         |  0
-------------------------------------------------
```

and then by the decomposition summary:

```
********* Multiplicity 1 Witness Set Decomposition *********

| dimension | components | classified | unclassified
-----------------------------------------------------
|   3       |   1        |   16       |  0
-----------------------------------------------------
```

```
************** Decomposition by Degree **************

Dimension 3: 1 classified component
-----------------------------------------------------
   degree 16: 1 component

*****************************************************
```

References

[1] D.J. Bates, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler, Bertini: Software for numerical algebraic geometry. Available at `bertini.nd.edu`.

[2] J.D. Hauenstein and C.W. Wampler. Unification and extension of intersection algorithms in numerical algebraic geometry. Available at `www.nd.edu/~jhauenst/preprints/hwGeneralIntersection.pdf`.