Tutorial 5: Variations on BBA

For a zero-dimensional polynomial ideal I, the **Border Basis Algorithm (BBA)** is an alternative to the Buchberger Algorithm for computing a border basis (and thus a Gröbner basis) of I. In this tutorial we invite you to implement it together with several variations and optimizations.

Let K be a field, let $P = K[x_1, \ldots, x_n]$, and let $I = \langle f_1, \ldots, f_s \rangle \subseteq P$ be a zero-dimensional polynomial ideal given by its generators $f_1, \ldots, f_s \in P \setminus \{0\}$. Moreover, let σ be a degree compatible term ordering. For a K-vector subspace of P, we let $V^+ = V + x_1V + \cdots + x_nV$.

The basic version of BBA is defined by the following instructions.

1. Let $V = \langle f_1, \ldots, f_s \rangle_K$ be the vector space spanned by $f_1, \ldots, f_s$.
2. Let $d = \max\{\deg(f_1), \ldots, \deg(f_s)\}$ and $U = \mathbb{T}^n_{\leq d}$.
3. Replace V by $V^+ \cap \langle U \rangle_K$ repeatedly until this procedure stabilizes.
4. Let $\mathcal{O}$ be the set of all terms in U which are not leading terms of polynomials in V.
5. If $\partial\mathcal{O}$ is not contained in U, increase d by one, let $U = \mathbb{T}^n_{\leq d}$, and continue with step (3).
6. Let $\mathcal{O} = \{t_1, \ldots, t_\mu\}$ and $\partial\mathcal{O} = \{b_1, \ldots, b_\nu\}$. For $j = 1, \ldots, \nu$, compute representations $b_j = \sum_{i=1}^{\mu} a_{ij}t_i + v_j$ with $a_{ij} \in K$ and $v_j \in V$.
7. For $j = 1, \ldots, \nu$, let $g_j = b_j - \sum_{i=1}^{\mu} a_{ij}t_i$. Then let $G = \{g_1, \ldots, g_\nu\}$. Return the pair $(\mathcal{O}, G)$.

In Theorem 6.4.36 of *the book* it is shown that this is an algorithm which computes a pair $(\mathcal{O}, G)$ such that $\mathcal{O}$ is the order ideal $\mathcal{O} = \mathbb{T}^n \setminus \mathrm{LT}_\sigma\{I\}$ and G is the $\mathcal{O}$-border basis of I. The vector space $\langle U \rangle_K$ is called the **computational universe**. Since it determines the size of the matrices we have to deal with, it is essential to keep it as small as possible. The vector space for which the procedure $V \mapsto V^+ \cap \langle U \rangle_K$ stabilizes is called the **U-stable span** of V.

a) Show that one can compute the U-stable span of V in the following way. First the generators of V^+ are interreduced. Then $V^+ \cap \langle U \rangle_K$ is obtained by simply taking those basis elements of V^+ whose leading term is in U.

b) Implement the BBA in a CoCoA function `BBA(...)`. Use your function to compute the `DegRevLex`-border basis of the following ideals.

   1. $I_1 = \langle y^3 - y,\ xy^2 - xy,\ 2x^2 + 2xy - 2x - y^2 - y \rangle \subseteq \mathbb{Q}[x, y]$
   2. $I_2 = \langle z^2 + 3y - 7z,\ yz - 4y,\ xz - 4y,\ y^4 - 4y,\ xy - 4y,\ x^5 - 8x^4 + 14x^3 + 8x^2 - 15x + 15y \rangle \subseteq \mathbb{Q}[x, y, z]$
   3. $I_3 = \langle x^2 - xy + y^2,\ x^3 - x^2y,\ x^2y - xy^2,\ xy^2 - y^3,\ x^3 + y^3 \rangle \subseteq \mathbb{Q}[x, y]$
   4. The ideals $J_1 - J_{10}$ of Tutorial 1.

c) Next we try to improve the BBA. In order to keep the computational universe small, we replace steps (2), (3), and (5) by the following instructions.

2'. Let U be the order ideal generated by the terms in the supports of the polynomials $f_1, \ldots, f_s$, i.e. take all these terms and their divisors.

3'. Compute a K-basis of V^+ consisting of polynomials with pairwise different leading terms. Append the terms in the supports of the new polynomials and the divisors of these terms to U. Replace V by $V^+ \cap \langle U \rangle_K$ and repeat this procedure until it stabilizes.

5'. If $\partial\mathcal{O}$ is not contained in U, replace U by $U \cup \partial U$ and continue with step (3').

$(*)$ Show that the modified procedure is still an algorithm which computes the $\mathcal{O}_\sigma(I)$-border basis of I.

d) Implement the **Improved Border Basis Algorithm (IBBA)** in a CoCoA function `IBBA(...)`. Then apply this function to the examples in (b). Protocol the dimensions of the computational universe for both BBA and IBBA in all cases.

e) The next optimization is of a slightly optimistic nature. We keep the enlargement of the computational universe in step (5) as small as possible by replacing it with

5". If $\partial\mathcal{O}$ is not contained in U, replace U by the order ideal generated by $U \cup \partial\mathcal{O}$ and continue with step (3').

$(**)$ Show that the modified procedure is still an algorithm which computes the $\mathcal{O}_\sigma(I)$-border basis of I. (This part has two stars, since we do not know how to prove termination.)

f) Implement the **Optimistic Border Basis Algorithm (OBBA)** in a CoCoA function `OBBA(...)`. Then apply this function to the examples in (b). Protocol the dimensions of the computational universe for OBBA in all cases and compare the result to your previous protocol. Moreover, keep track of the number of enlargements of U which are necessary.

g) The final version of BBA we develop is of a very experimental nature. We try to cut down on the number of iterations needed to find the U-stable span of V. For this we replace step (3') by the following instruction.

3". Enlarge V to a vector space V^{++} by adding for each basis element v of V all products tv such that $t \in \mathbb{T}^n$ and $\mathrm{LT}_\sigma(tv) \in U \cup \partial U$. Compute a K-basis of V^{++} consisting of polynomials with pairwise different leading terms. Append the terms in the supports of the new polynomials and the divisors of these terms to U. Replace V by $V^{++} \cap \langle U \rangle_K$ and repeat this procedure until it stabilizes.

$(**)$ Find out whether the modified procedure is still an algorithm which computes the $\mathcal{O}_\sigma(I)$-border basis of I.

h) Implement the **Experimental Border Basis Algorithm (EBBA)** in a CoCoA function `EBBA(...)`. Then apply this function to the examples in (b). Protocol the numbers of iterations of step (3) resp. (3") needed for BBA resp. EBBA in all cases.

Do we have to mention that there are also an Approximate Border Basis Algorithm (ABBA) and an Improved Approximate Border Basis Algorithm (IABBA)? Yabba Dabba Dooo!