

**Table 1** Differences between our pseudocode and that of Faugère and/or Stegers

<i>We change...</i>	<i>to...</i>	<i>because...</i>
Faugère, Stegers, Basis: compute Gröbner bases of $\langle f_m \rangle, \langle f_{m-1}, f_m \rangle, \dots, \langle f_1, f_2, \dots, f_m \rangle$ .	Compute Gröbner bases of $\langle f_1 \rangle, \langle f_1, f_2 \rangle, \dots, \langle f_1, f_2, \dots, f_m \rangle$ .	Personal preference; seems more natural.
Stegers, Critical_Pair: check wrt $G_{\nu_1+1}$ or $G_{\nu_2+1}$ .	Check top-reducible wrt $G_{\text{prev}}$ .	Agreement with Theorem ?? and Faugère.
Faugère, Critical_Pair: check $\text{Sig}(k) \prec \text{Sig}(\ell)$ .	Check $u_1 \cdot \text{Sig}(k) \prec u_2 \cdot \text{Sig}(\ell)$ .	Correct signature of $S$ -polynomial; agreement with Stegers.
Faugère and Stegers, Compute_SPols: do not order $P$ .	Order $P$ .	Performance; agreement with Faugère's example. If $P$ is not ordered, the algorithm generates 8 new polynomials, rather than the 7 documented by Faugère.
Stegers, Compute_SPols: Append $(u \cdot \text{Sig}(\ell), s)$ to $L$ .	Append $(u \cdot \text{Sig}(k), s)$ to $L$ .	Correct signature of $S$ -polynomial, agreement with Faugère.
Faugère and Stegers, Reduction: $\text{redo} \cup \text{to\_do}$ .	Insert elements of $\text{redo}$ into $\text{to\_do}$ by ascending signature.	$\text{to\_do}$ is a list, performance.
Faugère, Top_Reduction: <b>return</b> $\{\emptyset, \{k, j\}\}$ .	<b>return</b> $\{\emptyset, \{k, \#L\}\}$ .	Poly( $\#L$ ) was newly added and needs reduction, not Poly( $j$ ); agreement with Stegers.

## 1 Pseudocode

This section contains the pseudocode for F5. It is essentially the same as that described by Faugère and Stegers, with minor corrections indicated by comments. Table 1 summarizes these corrections.

We have implemented F5 in the interpreted language of the freely-available, open-source computer algebra system SINGULAR; interested readers will find the source code at

[http://www.math.usm.edu/perry/Research/f5\\_library.lib](http://www.math.usm.edu/perry/Research/f5_library.lib)

This implementation follows the pseudocode given here very closely. The user will also need to load the companion program

<http://www.math.usm.edu/perry/Research/f5ex.lib>

which contains a number of benchmark systems.

Albrecht has used our pseudocode to implement F5 in Sage [?], and King has done the same with Sage's Cython interface.

The implementations are primarily for study purposes: to examine a working implementation of F5 and experiment with it. It is much faster than an implementation of the Gebauer-Möller algorithm [?] in the interpreted language of SINGULAR, but nowhere as fast as the implementations of Faugère or Stegers. This is expected, however: Faugère's implementation consists of compiled, highly optimized C code, which would certainly be faster. Although Stegers' implementation consists of interpreted MAGMA code, the developers of SINGULAR inform us in private communication that the interpreted language of SINGULAR, unlike its kernel, is highly unoptimized. Work is in progress to implement F5 in the SINGULAR kernel. In addition, the efficiency of this implementation equals that of Stegers', as indicated by the number of  $S$ -polynomials that reduce to zero.

## References

**Algorithm 1** Basis

---

```

1: globals  $L, Rules, <_T$ 
2: inputs
3:    $F = (f_1, f_2, \dots, f_m) \in \mathcal{R}^m$  — homogeneous; if  $F$  is not homogeneous then homogenize
4:    $<$ , an admissible ordering
5: outputs
6:   a Gröbner basis of  $F$  with respect to  $<$ 
7: do
8:    $<_T := <$ 
9:   Sort  $F$  by increasing total degree, breaking ties by increasing leading monomial
   — Initialize the record keeping.
10:   $Rules := \text{Array}(1 \dots m)$ 
11:   $Rules_1 := \text{List}()$ 
12:   $L := \text{List}()$ 
   — One way to represent signatures would be as a tuple:  $\mu e_\nu \rightarrow (\nu, \mu)$ 
13:  Append  $(\mathbf{e}_1, f_1 \cdot (\text{lc}(f_1))^{-1})$  to  $L$ 
   — Compute the basis of  $(f_1)$ .
14:   $G_{\text{prev}} = \{1\}$ 
15:   $B = \{f_1\}$ 
   — Compute the bases of  $(f_1, f_2), \dots, (f_1, f_2, \dots, f_m)$ .
16:   $i := 2$ 
17:  while  $i \leq m$ 
18:     $G_{\text{curr}} := \text{Incremental\_Basis}(f_i, i, B, G_{\text{prev}})$ 
19:    if  $\exists \lambda \in G_{\text{curr}}$  such that  $\text{Poly}(\lambda) = 1$ 
20:      return  $\{1\}$ 
21:     $G_{\text{prev}} := G_{\text{curr}}$ 
22:     $B := \{f_\lambda : \lambda \in G_{\text{prev}}\}$ 
23:     $i := i + 1$ 
24:  return  $B$ 

```

---

**Algorithm 2** Incremental\_Basis

---

```

1: globals  $L, <_T$ 
2: inputs
3:    $f \in \mathcal{R}$ 
4:    $i \in \mathbb{N}$ 
5:    $B$ , a Gröbner basis of  $(f_1, f_2, \dots, f_{i-1})$  with respect to  $<_T$ 
6:    $G_{\text{prev}} \subset \mathbb{N}$ , indices in  $L$  of  $B$ 
7: outputs
8:    $G_{\text{curr}}$ , indices in  $L$  of a Gröbner basis of  $(f_1, f_2, \dots, f_i)$  with respect to  $<_T$ 
9: do
10:  Append  $(\mathbf{e}_i, f \cdot (\text{lc}(f))^{-1})$  to  $L$ 
11:   $\text{curr\_idx} := \#L$ 
12:   $G_{\text{curr}} := G_{\text{prev}} \cup \{\text{curr\_idx}\}$ 
13:   $Rules_i := \text{List}()$ 
14:   $P := \bigcup_{j \in G_{\text{prev}}} \text{Critical\_Pair}(\text{curr\_idx}, j, i, G_{\text{prev}})$ 
15:  while  $P \neq \emptyset$ 
16:     $d := \min \{\text{deg } t : (t, k, u, \ell, v) \in P\}$  — See Algorithm 3 for structure of  $p \in P$ 
17:     $P_d := \{(t, k, u, \ell, v) \in P : d = \text{deg } t\}$ 
18:     $P := P \setminus P_d$ 
19:     $S := \text{Compute\_SPols}(P_d)$ 
20:     $R := \text{Reduction}(S, B, G_{\text{prev}}, G_{\text{curr}})$ 
21:    for  $k \in R$ 
22:       $P := P \cup \left( \bigcup_{j \in G_{\text{curr}}} \text{Critical\_Pair}(j, k, i, G_{\text{prev}}) \right)$ 
23:       $G_{\text{curr}} := G_{\text{curr}} \cup \{k\}$ 
24:  return  $G_{\text{curr}}$ 

```

---

---

**Algorithm 3** Critical\_Pair

---

```

1: globals  $\prec_T$ 
2: inputs
3:    $k, \ell \in \mathbb{N}$  such that  $1 \leq k < \ell \leq \#L$ 
4:    $i \in \mathbb{N}$ 
5:    $G_{\text{prev}} \subset \mathbb{N}$ , indices in  $L$  of a Gröbner basis of  $(f_1, f_2, \dots, f_{i-1})$  w/respect to  $\prec_T$ 
6: outputs
7:    $\{(t, u, k, v, \ell)\}$ , corresponding to a critical pair  $\{k, \ell\}$  necessary for
8:     the computation of a Gröbner basis of  $(f_1, f_2, \dots, f_i)$ ;  $\emptyset$  otherwise
9: do
10:   $t_k := \text{lm}(\text{Poly}(k))$ 
11:   $t_\ell := \text{lm}(\text{Poly}(\ell))$ 
12:   $t := \text{lcm}(t_k, t_\ell)$ 
13:   $u_1 := t/t_k$ 
14:   $u_2 := t/t_\ell$ 
15:   $\mu_1 \mathbf{e}_{\nu_1} := \text{Sig}(k)$ 
16:   $\mu_2 \mathbf{e}_{\nu_2} := \text{Sig}(\ell)$ 
17:  if  $\nu_1 = i$  and  $u_1 \cdot \mu_1$  is top-reducible by  $G_{\text{prev}}$  — Stegers checks by  $G_{\nu_1+1}$ 
18:    return  $\emptyset$ 
19:  if  $\nu_2 = i$  and  $u_2 \cdot \mu_2$  is top-reducible by  $G_{\text{prev}}$  — Stegers checks by  $G_{\nu_2+1}$ 
20:    return  $\emptyset$ 
    — A minor optimization is to check  $\text{Is\_Rewritable}$  here
21:  if  $u_1 \cdot \text{Sig}(k) \prec u_2 \cdot \text{Sig}(\ell)$  — Faugère's writeup compares  $\text{Sig}(k) \prec \text{Sig}(\ell)$ .
22:    Swap  $u_1$  and  $u_2$ 
23:    Swap  $k$  and  $\ell$ 
24:  return  $\{(t, k, u_1, \ell, u_2)\}$ 

```

---



---

**Algorithm 4** Compute\_SPols

---

```

1: globals  $L, \prec_T$ 
2: inputs
3:    $P$ , a set of critical pairs in the form  $(t, k, u, \ell, v)$ 
4: outputs
5:    $S$ , a list of indices in  $L$  of  $S$ -polynomials computed
6:   for a Gröbner basis of  $(f_1, f_2, \dots, f_i)$ 
7: do
8:    $S := ()$ 
    — Faugère and Stegers do not indicate that one should sort  $P$ , but performance suffers if not.
    — For the example in Faugère's paper, 8 polynomials would be computed, not 7.
9:   for  $(t, k, u, \ell, v) \in P$ , from smallest to largest lcm
10:    if not  $\text{Is\_Rewritable}(u, k)$  and not  $\text{Is\_Rewritable}(v, \ell)$ 
11:      Compute  $s$ , the  $S$ -polynomial of  $\text{Poly}(k)$  and  $\text{Poly}(\ell)$ 
12:      if  $s \neq 0$ 
13:        Append  $(u \cdot \text{Sig}(k), s)$  to  $L$  — Stegers writes  $\text{Sig}(\ell)$ .
14:        Add_Rule( $u \cdot \text{Sig}(k)$ ,  $\#L$ )
15:        Append  $\#L$  to  $S$ 
16:   Sort  $S$  by increasing signature
17:   return  $S$ 

```

---

**Algorithm 5** Reduction

---

```

1: globals  $L, <_T$ 
2: inputs
3:    $S$ , a list of indices of polynomials added to the generators  $G_i$ 
4:    $B$ , a Gröbner basis of  $(f_1, f_2, \dots, f_{i-1})$  with respect to  $<_T$ 
5:    $G_{\text{prev}} \subset \mathbb{N}$ , indices in  $L$  corresponding to  $B$ 
6:    $G_{\text{curr}} \subset \mathbb{N}$ , indices in  $L$  of a list of generators of the ideal of  $(f_1, f_2, \dots, f_i)$ 
7: outputs
8:    $\text{completed}$ , a subset of  $G$  corresponding to (mostly) top-reduced polynomials
9: do
10:   $to\_do := S$ 
11:   $\text{completed} := \emptyset$ 
12:  while  $to\_do \neq ()$ 
13:    Let  $k$  be the element of  $to\_do$  such that  $\text{Sig}(k)$  is minimal.
14:     $to\_do := to\_do \setminus \{k\}$ 
15:     $h := \text{Normal\_Form}(\text{Poly}(k), B, <_T)$ 
16:     $L_k := (\text{Sig}(k), h)$ 
17:     $\text{newly\_completed}, \text{redo} := \text{Top\_Reduction}(k, G_{\text{prev}}, G_{\text{curr}} \cup \text{completed})$ 
18:     $\text{completed} := \text{completed} \cup \text{newly\_completed}$ 
    — Faugère and Stegers both write  $to\_do := to\_do \cup \text{redo}$ ,
    — but  $to\_do$  is not a set, and for efficiency needs to be sorted.
19:    for  $j \in \text{redo}$ 
20:      Insert  $j$  in  $to\_do$ , sorting by increasing signature
21:  return  $\text{completed}$ 

```

---

**Algorithm 6** Top\_Reduction

---

```

1: globals  $L, <_T$ 
2: inputs
3:    $k$ , the index of a labeled polynomial
4:    $G_{\text{prev}} \subset \mathbb{N}$ , indices in  $L$  of a Gröbner basis of  $(f_1, f_2, \dots, f_{i-1})$  w/respect to  $<_T$ 
5:    $G_{\text{curr}} \subset \mathbb{N}$ , indices in  $L$  of a list of generators of the ideal of  $(f_1, f_2, \dots, f_i)$ 
6: outputs
7:    $\text{completed}$ , which has value  $\{k\}$  if  $L_k$  was not top-reduced and  $\emptyset$  otherwise
8:    $to\_do$ , which has value
9:      $\emptyset$  if  $L_k$  was not top-reduced,
10:     $\{k\}$  if  $L_k$  is replaced by its top-reduction, and
11:     $\{k, \#L\}$  if top-reduction of  $L_k$  generates a polynomial with a signature larger than  $\text{Sig}(k)$ .
12: do
13:  if  $\text{Poly}(k) = 0$  — This condition should be false if the inputs are a regular sequence.
14:    warn “Reduction to zero!”
15:    return  $\emptyset, \emptyset$ 
16:   $p := \text{Poly}(k)$ 
17:   $J := \text{Find\_Reductor}(k, G_{\text{prev}}, G_{\text{curr}})$ 
18:  if  $J = \emptyset$ 
19:     $L_k := (\text{Sig}(k), p \cdot (\text{lc}(p))^{-1})$ 
20:    return  $\{k\}, \emptyset$ 
    —  $J \neq \emptyset$ , so it is safe to top-reduce.
21:  Let  $j$  be the single element in  $J$ 
22:   $q := \text{Poly}(j)$ 
23:   $u := \frac{\text{lm}(p)}{\text{lm}(q)}$ 
24:   $c := \text{lc}(p) \cdot (\text{lc}(q))^{-1}$ 
25:   $p := p - c \cdot u \cdot q$ 
26:  if  $p \neq 0$ 
27:     $p := p \cdot (\text{lc}(p))^{-1}$ 
28:  if  $u \cdot \text{Sig}(j) \prec \text{Sig}(k)$ 
29:     $L_k := (\text{Sig}(k), p)$ 
30:    return  $\emptyset, \{k\}$ 
31:  else
32:    Append  $(u \cdot \text{Sig}(j), p)$  to  $L$ 
33:    Add_Rule  $(u \cdot \text{Sig}(j), \#L)$ 
    — Faugère writes  $\emptyset, \{k, j\}$  below, but  $\text{Poly}(\#L)$  needs top-reduction, not  $\text{Poly}(j)$ .
34:    return  $\emptyset, \{k, \#L\}$ 

```

---

---

**Algorithm 7** Find\_Reductor

---

```

1: globals  $<_T$ 
2: inputs
3:    $k$ , the index of a labeled polynomial
4:    $G_{\text{prev}} \subset \mathbb{N}$ , indices in  $L$  of a Gröbner basis with respect to  $<_T$  of  $(f_1, f_2, \dots, f_{i-1})$ 
5:    $G_{\text{curr}} \subset \mathbb{N}$ , indices in  $L$  of a list of generators of the ideal of  $(f_1, f_2, \dots, f_i)$ 
6: outputs
7:    $J$ , where  $J = \{j\}$  if  $j \in G_{\text{curr}}$  and  $\text{Poly}(k)$  is safely top-reducible by  $\text{Poly}(j)$ ;
8:   otherwise  $J = \emptyset$ 
9: do
10:   $t := \text{lm}(\text{Poly}(k))$ 
11:  for  $j \in G_{\text{curr}}$ 
12:     $t' = \text{lm}(\text{Poly}(j))$ 
13:    if  $t' \mid t$ 
14:       $u := t/t'$ 
15:       $\mu_j \mathbf{e}_{\nu_j} := \text{Sig}(j)$ 
16:      if  $u \cdot \text{Sig}(j) \neq \text{Sig}(k)$  and not  $\text{Is\_Rewritable}(u, j)$  and  $u \cdot \mu_j$  is not top-reducible by  $G_{\text{prev}}$ 
17:        return  $\{j\}$ 
18:  return  $\emptyset$ 

```

---



---

**Algorithm 8** Add\_Rule

---

```

1: globals  $L, Rules$ 
2: inputs
3:    $\mu \mathbf{e}_{\nu}$ , the signature of  $L_k$ 
4:    $k$ , the index of a labeled polynomial in  $L$  (or 0, for a phantom labeled polynomial)
5: do
6:   Append  $(\mu, k)$  to  $Rules_{\nu}$ 
7:   return

```

---



---

**Algorithm 9** Is\_Rewritable

---

```

1: inputs
2:    $u$ , a power product
3:    $k$ , the index of a labeled polynomial in  $L$ 
4: outputs
5:   true if  $u \cdot \text{Sig}(k)$  is rewritable by another labeled polynomial (see Find_Rewriting)
6: do
7:    $j := \text{Find\_Rewriting}(u, k)$ 
8:   return  $j \neq k$ 

```

---



---

**Algorithm 10** Find\_Rewriting

---

```

1: globals  $Rules$ 
2: inputs
3:    $u$ , a power product
4:    $k$ , the index of a labeled polynomial in  $L$ 
5: outputs
6:    $j$ , the index of a labeled polynomial in  $L$  such that if  $\mu_j \mathbf{e}_{\nu_j} = \text{Sig}(j)$ 
   and  $\mu_j \mathbf{e}_{\nu_j} = \text{Sig}(k)$ , then  $\nu_j = \nu_k$  and  $\mu_j \mid u \cdot \mu_k$ 
   and  $L_j$  was added to  $Rules_{\nu_k}$  most recently.
7: do
8:    $\mu_k \mathbf{e}_{\nu} := \text{Sig}(k)$ 
9:    $ctr := \#Rules_{\nu}$ 
10:  while  $ctr > 0$ 
11:     $(\mu_j, j) := Rules_{\nu, ctr}$ 
12:    if  $\mu_j \mid u \cdot \mu_k$ 
13:      return  $j$ 
14:     $ctr := ctr - 1$ 
15:  return  $k$ 

```

---