The density method and permutations with a prescribed descent set

Philippe Marchal

LAGA (UMR CNRS 7539), Université Paris 13
marchal@math.univ-paris13.fr

Abstract

We give a formula to compute the number of permutations with a prescribed descent set in quadratic time. We give the generating function of the number of permutations with a periodic descent set. We introduce the *density method algorithm*, which generates uniformly distributed random permutations with a prescribed descent set.

1 Statement of the results

Let σ be permutation of $\{1, 2, \dots N\}$. The descent set of σ, $D(\sigma)$, is the set of integers $i \in [1, N-1]$ such that $\sigma(i) > \sigma(i+1)$. For instance, $D(id) = \emptyset$. If σ is an alternating permutation, that is, if

$$\sigma(1) > \sigma(2) < \sigma(3) > \sigma(4) \dots$$

then $D(\sigma) = [1, N-1] - 2\mathbb{N}$.

The aim of this paper is to study permutations of $\{1, 2, \dots N\}$ with a prescribed descent set $A \subset [1, N-1]$. We first give a method to compute this number efficiently, namely, using $O(N^2)$ elementary computations, while the classical formula based on an inclusion-exclusion argument uses an exponential number of elementary computations. Next, given a periodic pattern of ascents and descents, we give a complete expression of the exponential generating function of permutations whose descent set follows this pattern. Finally, we introduce an algorithm generating at random, with uniform probability, permutations with a given descent set.

These results make use of a generic method, which we call the *density method*, and which applies in a more general framework, namely the study of (finite) posets. Generally speaking, if Pos is a finite poset, its *order polytope* is the subset of $[0,1]^{Pos}$ consisting of all functions $f : Pos \to [0,1]$ satisfying, for all $x, y \in Pos$, $x \leq y \implies f(x) \leq f(y)$. The density method consists in expressing the marginal densities of the uniform measure on the order polytope by computing iterated integrals. This is not always possible for all types of posets but in the context of this paper, we shall see that this is relevant. An interest of this method is that it provides an explicit algorithm of random generation with polynomial cost. Other uses of this method can be found in [Mar16, BMW18].

Our first theorem is the following:

Theorem 1.1 *Fix an integer $N \geq 2$ and a set $A \subset [1, N-1]$. Define by descending induction a sequence of polynomials $(f_i, 1 \leq i \leq N)$ as follows:*

- $f_N = 1$

Copyright © by the paper's authors. Copying permitted for private and academic purposes.

In: L. Ferrari, M. Vamvakari (eds.): Proceedings of the GASCom 2018 Workshop, Athens, Greece, 18–20 June 2018, published at http://ceur-ws.org

CEUR-WS.org/Vol-2113/paper19.pdf

- *If $i \in A$,*
$$f_i(x) = \int_0^x f_{i+1}(y)\, dy$$
- *If $i \notin A$,*
$$f_i(x) = \int_x^1 f_{i+1}(y)\, dy$$

Let $Q_N(A)$ be the number of permutations of $\{1, 2, \dots N\}$ with descent set A. Then

$$Q_N(A) = N! \int_0^1 f_1(y)\, dy$$

The reason why we use descending induction, rather than regular induction, will appear in Theorem 3. An easy inclusion-exclusion argument (see for instance [Bon12], Chapter 1) gives

$$Q_N(A) = \sum_{B \subset A} (-1)^{|A|-|B|} g_N(B) \tag{1}$$

where the function g_N is defined as follows. If $B = \emptyset$, then $g_N(B) = 1$. Otherwise, there exists $k \in [1, N-1]$ such that B has the form $B = \{i_1 < i_2 < \dots < i_k\}$ and in that case

$$g_N(B) = \frac{N!}{(N-i_k)!(i_k - i_{k-1})! \dots (i_2 - i_1)! i_1!}$$

However, in order to use (1), one needs to compute $2^{N-|A|}$ terms. By comparison, Theorem 1 only requires the computations of the coefficients of N polynomials of degree 0 to $N-1$, which means computing $O(N^2)$ coefficients. Moreover, each coefficient can be computed using only one elementary computation, except for the constant coefficients. It is easily seen that calculating the constant coefficient of f_d requires $N-d$ elementary computations. Hence the number of elementary computations in Theorem 1 is $O(N^2)$. Another efficient method to compute Q_N is given by Viennot [Vie79].

Theorem 1 also gives access to comparison results. For a permutation σ of $[1, N]$, say that $m \in [2, N-1]$ is a local extremum if either $\sigma(m-1) < \sigma(m) > \sigma(m+1)$ or $\sigma(m-1) > \sigma(m) < \sigma(m+1)$. Then we have

Corollary 1.1 *For $B \subset [2, N-1]$, let $F(B)$ be the number of permutations of $[1, N]$ with set of local extrema B. Then $F : \mathcal{P}(\{2, 3, \dots n-1\}) \to \mathbb{N}$ is an increasing function.*

Of course, such results are more difficult to derive using alternating sums such as (1). The fact that the maximum of F is achieved for alternating permutations was first observed by Niven [Niv68], see also [BR70]. Our next result deals with the case of a periodic descent set.

Theorem 1.2 *Let $p \geq 2$ be an integer, fix a set $A \subset \{1, 2, \dots p\}$ and put*

$$A_\infty = \bigcup_{n=0}^{\infty} (A + np)$$

For each integer $n \geq 1$, let d_n be the number of permutations of $\{1, \dots, n\}$ with descent set

$$A_\infty \cap [1, n-1]$$

Let $\omega_1, \dots, \omega_p$ be the p-th roots of 1 (resp. -1) if $p - |A|$ is even (resp. odd). For $k, l \in [1, p]$, put

$$g_{k,l}(t) = \sum_{n=0}^{\infty} \frac{(\omega_k t)^{np+l}}{(np+l)!}$$

Then there exists a rational function $R_A \in \mathbb{Z}(X_1, \dots, X_{p(p+1)})$, such that

$$\sum_{n=1}^{\infty} \frac{d_n t^n}{n!} = R_A(\omega_1, \dots, \omega_p, g_{1,1}(t), \dots, g_{p,p}(t))$$

Theorem 2 generalizes the classical theorem by André [And1879] for alternating permutations:

$$\sum_{n=0}^{\infty} \frac{d_n t^n}{n!} = \frac{1+\sin(t)}{\cos(t)}$$

where we agree that $d_0 = 1$. More recent results were established by Carlitz (see [Car75] for instance) and Mendes-Remmel-Riehl [MRR10] in the case $\mu = 1$, and later by Luck [Luc14] and Basset [Bas16]. We shall see how to recover these results from our method in Section 3.

For the first-order asymptotics, see [LM83] and [BHR03]. For every set A, one can compute R_A and derive the asymptotics from the zeros of the denominator, using the classical tools of analytic combinatorics. See for instance the analysis of alternating permutations as Example IV.35 in [FS09]. However, our approach does not provide the general form of the denominator.

Finally, we introduce an algorithm generating uniform random permutations with a given descent set. We begin by a simple remark, which was already used in [ELR02] among others. Let $Y = (Y_1, \dots Y_N)$ be a sequence of distinct reals in $[0,1]$. We can construct from Y a permutation σ_Y as follows. Let $k_1 \in \{1, 2, \dots N\}$ be the integer such that Y_{k_1} is minimal in $\{Y_1, \dots Y_N\}$ and put $\sigma_Y(k_1) = 1$. Then, let $k_2 \in \{1, 2, \dots N\} - \{k_1\}$ be the integer such that Y_{k_2} is minimal in $\{Y_1, \dots Y_N\} - \{Y_{k_1}\}$, put $\sigma_Y(k_2) = 2$ and so on. To recover σ_Y from Y, one can use a sorting algorithm.

One can define the descent set $D(Y)$ of a sequence of reals Y as for a permutation and clearly, $D(Y) = D(\sigma_Y)$. Moreover, if Y is chosen according to the Lebesgue measure on $[0,1]^N$, then σ_Y is uniformly distributed over all permutations of $\{1, 2, \dots N\}$. As a consequence, if Y is chosen uniformly at random among all sequences with descent set A, that is, if its density with respect to the Lebesgue measure on $[0,1]^N$ is

$$C\mathbf{1}_{\{D(Y)=A\}} \tag{2}$$

for some constant $C > 0$, then σ_Y is uniformly distributed over all permutations with descent set A. Therefore, we want to find an algorithm constructing a random sequence with descent set A.

Algorithm Using iid random variables $(U_1, \dots U_N)$, uniform on $[0,1]$, and the functions f_i defined in Theorem 1, construct a sequence $Y = (Y_1, \dots Y_N)$ as follows:

- Y_1 is the real in $[0,1]$ such that
$$\int_0^{Y_1} f_1(y)dy = U_1 \int_0^1 f_1(y)\, dy$$
- For $i \in [1, N-1]$, Y_{i+1} is the only solution in $[0,1]$ of the equation
$$f_i(Y_{i+1}) = U_{i+1} f_i(Y_i) \tag{3}$$

Finally, recover the permutation σ_Y from Y by sorting.

Theorem 1.3 *The algorithm described above yields a random permutation of $\{1, 2, \dots N\}$, uniformly distributed over all permutations with descent set A.*

To see that the algorithm is well-defined, suppose for instance that $i \in A$. Then (3) becomes

$$\int_0^{Y_{i+1}} f_{i+1}(y)\, dy = U_{i+1} f_i(Y_i) \tag{4}$$

Since the functions f_i, f_{i+1} are nonnegative on the interval $[0,1]$, (4) clearly has a unique solution on $[0,1]$.

We shall not study in detail the complexity of the algorithm. Nevertheless, let us make a few comments.

First, remark that a simple rejection algorithm would have exponential complexity. Indeed, if we draw a permutation at random, the most likely profile is the alternating case, and the probability for a random permutation of length N to be alternating is $\sim \frac{4}{\pi}(2/\pi)^N$ (see Example IV.35 in [FS09]). Therefore, the average number of times one has to draw a permutation before finding one with the prescribed descent set is at least $c(\pi/2)^N$.

Using our algorithm, we need to compute the functions f_i, which can be done using $O(N^2)$ elementary computations, as already seen. The last step, namely sorting the sequence Y to deduce σ_Y, has an average complexity $O(N \log N)$ if one uses randomized quicksort. What is more intricate is to evaluate the time needed to generate the variables Y_i, which amounts to solving N equations of the form (4). We shall not discuss this from a theoretical point of view. However, some experimental results are given at the end of the paper. Typically, our algorithm can generate permutations of length a few thousands.

In the particular case of alternating permutations, other algorithms with a better complexity can be found in [BRS12].

We now proceed to the proof of the results. We first prove Theorem 3 in Section 2 and deduce Theorem 1 and Corollary 1 in Section 3. In Section 4, we give an outline of the proof of Theorem 2. In the last section , we comment on the implementation in Maple of our algorithm.

2 Proof Theorem 1.3

First, observe that there are obvious symmetries in the problem. If one replaces $\sigma(i)$ with $N+1-\sigma(i)$ for each i, then A is replaced with its complement. If $\sigma(i)$ is replaced with $\sigma(N+1-i)$ for each i, then A is replaced with $N-A$.

Let us prove Theorem 1.3. One can re-formulate the algorithm, using the notion of conditional density of a random variable, as follows:

- Y_1 has density $f_1/\int_0^1 f_1(y)\,dy$.
- If $i \in A$, conditionally on Y_i, Y_{i+1} has density $\mathbf{1}_{[0,Y_i]} f_{i+1}/f_i(Y_i)$.
- If $i \notin A$, conditionally on Y_i, Y_{i+1} has density $\mathbf{1}_{[Y_i,1]} f_{i+1}/f_i(Y_i)$.

As a consequence, the density of the sequence $Y = (Y_1, \ldots, Y_N)$ with respect to the Lebesgue measure on $[0,1]^N$ is equal to the telescopic product

$$\begin{aligned}&\frac{f_1(Y_1)}{\int_0^1 f_1(y)\,dy} \times \frac{f_2(Y_2)}{f_1(Y_1)} \times \ldots \times \frac{f_N(Y_N)}{f_{N-1}(Y_{N-1})} \mathbf{1}_{\{D(Y)=A\}} \\ &= \frac{1}{\int_0^1 f_1(y)\,dy} \mathbf{1}_{\{D(Y)=A\}}\end{aligned}$$

which is exactly the form required in (2). Therefore, σ_Y is uniformly distributed over all permutations with descent set A. This proves Theorem 3.

3 Proof of Theorem 1.1

Let us deduce Theorem 1. If a permutation of $\{1, \ldots n\}$ is drawn uniformly at random, the probability that its descent set is A is

$$\int_0^1 dy_1 \ldots \int_0^1 dy_N \mathbf{1}_{\{D(Y)=A\}}$$

But this probability is also equal to

$$\frac{Q_N(A)}{N!}$$

Using the formula for the density of Y, we find that

$$\frac{1}{\int_0^1 f_1(y)dy} \int_0^1 dy_1 \ldots \int_0^1 dy_N \mathbf{1}_{\{D(y)=A\}} = 1$$

whence

$$Q_N(A) = N! \int_0^1 f_1(y)\,dy$$

which proves Theorem 1.

Finally, let us prove Corollary 1. It suffices to prove that if $B, \tilde{B}$ are two subsets of $[2, N-1]$ with $\tilde{B} = B \cup \{m\}$, $m \notin B$, then the number of permutations with set of extrema B is smaller than the number of permutations with

set of extrema $\tilde{B}$. Using the symmetries of the problem, it suffices to count permutations with set of extrema $B, \tilde{B}$ ending with a descent. If we impose this condition, let $A, \tilde{A}$ be the corresponding descent sets. Using $A, \tilde{A}$ we define by descending induction the polynomials $f_i, \tilde{f}_i$ as in Theorem 1.

Now define the polynomials h_i, $1 \leq i \leq N$ by $h_N = 1$ and $h_i(x) = f_i(x)$ if $i \in A$ while $h_i(x) = f_i(1-x)$ if $i \notin A$. Remark that for $i \leq N-1$, the h_i are increasing functions on $[0,1]$ such that $h_i(0) = 0$. Besides, one can directly define the h_i by descending induction as follows. First, $h_N = 1$ and then, if $i \notin B$,

$$h_i(x) = \int_0^x h_{i+1}(y)\, dy \tag{5}$$

while if $i \in B$,

$$h_i(x) = \int_0^x h_{i+1}(1-y)\, dy \tag{6}$$

Likewise, define the polynomials $\tilde{h}_i$, $1 \leq i \leq N$ by $\tilde{h}_N = 1$, $\tilde{h}_i(x) = \tilde{f}_i(x)$ if $i \in \tilde{A}$ and $\tilde{h}_i(x) = \tilde{f}_i(1-x)$ if $i \notin \tilde{A}$.

Since $\tilde{B} = B \cup \{m\}$, we have $\tilde{h}_i = h_i$ for $i > m$. Moreover, using (5) and (6), we get that $\tilde{h}_m(x) > h_m(x)$ for every $x \in (0,1]$ and by induction, for every $i < m$ and every $x \in (0,1]$, $\tilde{h}_i(x) > h_i(x)$. Integrating this inequality for $i = 1$, we get the result.

Remark 3.1 The coefficients of the polynomials f_i can be computed by induction. Put

$$f_{N-i}(x) = \sum_{k=0}^{i} a_k^{(i)} x^k$$

First, $a_0^{(0)} = 1$. Next, if $N - i \in A$, then $a_0^{(i+1)} = 0$ and for $k \geq 0$,

$$a_{k+1}^{(i+1)} = \frac{a_k^{(i)}}{k+1}$$

On the other hand, if $N - i \notin A$, then for $k \geq 0$,

$$a_{k+1}^{(i+1)} = -\frac{a_k^{(i)}}{k+1}$$

and

$$a_0^{(i+1)} = \sum_{k=0}^{i} \frac{a_k^{(i)}}{k+1}$$

Therefore, if $N - i \notin A$,

$$a_0^{(i+1)} = \sum_{j=0}^{i} \frac{a_0^{(i-j)}}{(j+1)!} (-1)^{|A^c \cap [N-i+1, N-1]|}$$

By recursion, this leads to (1), which provides an alternative proof of Theorem 1.

4 Proof of Theorem 1.2 (outline)

Define by induction the sequence of polynomials (f_n) by

- $f_0 = 1$
- If $i \in A_\infty$, $i \geq 1$
$$f_i(x) = \int_0^x f_{i-1}(y) dy$$
- If $i \notin A_\infty$, $i \geq 1$
$$f_i(x) = \int_x^1 f_{i-1}(y) dy$$

We claim that

$$d_N = N! \int_0^1 f_{N-1}(x)dx$$

Indeed, Theorem 1 tells us that the right-hand side is the number of permutations with length N and descent set $N - D_N$. Using the symmetries of the problem, we see that this is equal to the number of permutations with length N and descent set D_N.

Consider the bivariate generating function

$$F(x,t) = \sum_{n=0}^{\infty} t^n f_n(x)$$

Then we have

$$\sum_{n=1}^{\infty} \frac{d_n}{n!} t^n = \int_0^1 tF(x,t)dx$$

We want to prove that the right-hand side of this equality has the form given by Theorem 2. Let μ be the number of ascents in a period, $\mu = p - |A|$. By differentiating,

$$\frac{\partial^p F(x,t)}{\partial x^p} = (-1)^\mu t^p F(x,t)$$

Solutions of this differential equation have the general form

$$F(x,t) = \sum_{k=1}^{p} a_k(t) e^{\omega_k t x}$$

where the ω_k are the p-th roots of 1 if μ is even and the p-th roots of -1 if μ is odd, and where the a_k are power series:

$$a_k(t) = \sum_{n \geq 0} a_{k,n} t^n$$

Therefore we can rewrite

$$\int_0^1 tF(x,t)dx = \sum_{k=1}^{p} \sum_{n \geq 0} a_{k,n} t^n \left(\frac{e^{\omega_k t} - 1}{\omega_k} \right)$$

We can also re-express

$$f_n(x) = \sum_{l=0}^{n} \sum_{k=1}^{p} \frac{\omega_k^l x^l}{l!} a_{k,n-l}$$

and so

$$f'_{n+1}(x) = \sum_{l=0}^{n} \sum_{k=1}^{p} \frac{\omega_k^{l+1} x^l}{l!} a_{k,n-l}$$

It is thus possible to compute the the coefficients $a_{k,n}$ using the fact that

- If $n \geq 1$ is a descent, $f_n(0) = 0$ whence

  $$\sum_{k=1}^{p} a_{k,n} = 0$$

  and moreover $f'_{n+1} = -f_n$, whence

  $$\sum_{k=1}^{p} a_{k,n} \omega_k^j = \sum_{k=1}^{p} a_{k,n} \omega_k^{j+1}$$

- If $n \geq 1$ is an ascent, $f_n(0) = 1$ whence

  $$\sum_{k=1}^{p} a_{k,n} = -\sum_{j=1}^{n} \sum_{k=1}^{p} \frac{a_{k,n-j} \omega_k^j}{j!}$$

and moreover $f'_{n+1} = f_n$, whence

$$\sum_{k=1}^{p} a_{k,n}\omega_k^j = -\sum_{k=1}^{p} a_{k,n}\omega_k^{j+1}$$

All these equations can be summarized in a matrix system with integer coefficients. We skip the details of the resolution of this system (this is a bit long but only uses elementary linear algebra) but eventually we find

$$\sum_{n=1}^{\infty} \frac{d_n}{n!}t^n = \int_0^1 tF(x,t)dx = \sum_{k=1}^{p}\sum_{l=1}^{p} h_{k,l}(t)\frac{e^{\omega_k t}-1}{\omega_k} \tag{7}$$

where, for all $k, l \in [1, p]$, the function

$$h_{k,l}(t) = \sum_{n=0}^{\infty} a_{k,np+l}t^{np+l}$$

has a rational expression, with integer coefficients, in terms of the ω_k and of the functions

$$g_{k,l}(t) = \sum_{n=0}^{\infty} \frac{(\omega_k t)^{np+l}}{(np+l)!}$$

This proves Theorem 2.

5 Implementation

We have implemented our algorithm in Maple[1]. The length of the permutation is $nmax$. The random variables $U[n]$, which are drawn independently at random, represent the descent profile: $U[n] = 0$ or 1 according as whether $nmax - n$ is a descent or an ascent. The reals $V[n]$ in the code correspond to the random variables Y_i in the algorithm.

As pointed out in the introduction, generating the variables Y_i amounts to solving N equations of the form (4). Of course, we can only get approximate solutions and since the Y_i are computed recursively, there is a propagation of errors. We shall not discuss this point from a theoretical point of view here. Observe, however, that these errors do not affect the permutation we generate as long as, for each i, the difference between Y_i and its approximation is smaller than the minimal value of $2|Y_i - Y_j|$ over all $i, j \in [1, nmax]$.

Since equations of the form (4) are polynomial and since $f'_i = \pm f_{i-1}$, we can directly implement Newton's method in the algorithm. We stop the iterations in Newton's method as soon as two successive approximations of the solution are at distance less that 10^{-p}, where p is a parameter that can be tuned. We give the time needed to generate permutations of variable length and for various values of p. For a permutation of length 2000, we have run the algorithm for the values $p = 4$ and $p = 5$ and we observe empirically that this yields the same permutation.

Here are some tables giving the run time of the algorithm for permutations of various lengths. In the first table, we have also let the parameter p vary. The first line is the length of the permutation, the second line the run time (in seconds) for p=4 and the third line the run time for p=5.

100	200	300	400	500	600	700	800	900	1000
0.18	0.93	2.21	4.46	6.75	11.36	16.14	23.36	31.73	37.93
0.19	0.97	2.52	4.65	7.36	12.29	17.70	24.66	34.59	41.15

In the next simulations, we have taken p=4 and we only give the integer part of the run time (in seconds).

1200	1400	1600	1800	2000	2200	2400	2600	2800	3000
68	90	119	174	213	269	351	627	678	766

3300	3600	3900	4200	4500	4800	5100	5400	5700	6000
840	1210	1410	1822	2385	3283	5395	5877	6380	9121

Empirically, the run time is more than quadratic. This is due to the fact that we have to perform computations with polynomials of large degree. Also, the quantity of memory is quite large if we want to record all the coefficients of all the polynomials. Of course, only keeping in memory the polynomial of highest degree and the

[1]The code is available at http://www.math.univ-paris13.fr/∼marchal/algopermut.m

descent profile would be sufficient, since the other polynomials can be obtained by differentiation. However, this would mean computing these polynomials twice.

Acknowledgements

I am very grateful to Cyril Banderier for his help in the implementation of the algorithm. I also thank Olivier Bodini and Michael Wallner for useful comments and references. This work is partially supported by the research project "Combinatoire à Paris".

References

[And1879] D. André. Développements de $\sec x$ et $\tan x$. *Comptes Rendus de l'Académie des Sciences*, Paris, 88:965-967, 1879.

[BMW18] C. Banderier, P. Marchal, W. Michael. Rectangular Young tableaux with local decreases and the density method for uniform random generation. *Preprint*, 2018.

[BHR03] E. A Bender, W. J. Helton, L.B. Richmond. Asymptotics of permutations with nearly periodic patterns of rises and falls. *Electronic Journal of Combinatorics*, 10, 2003.

[Bas16] N. Basset. Counting and generating permutations in regular classes. *Algorithmica*, 76:989-1034, 2016.

[Bon12] M. Bóna. *Combinatorics of permutations.* Second edition. Series: Discrete Mathematics and its Applications (Boca Raton). CRC Press, Boca Raton, FL, 2012.

[BRS12] O. Bodini, O. Roussel, M. Soria. Boltzmann samplers for first-order differential specifications. *Discrete Applied Mathematics*, 160 (18):2563-2572, 2012.

[BR70] N. G.de Bruijn. Permutations with given ups and downs. *Nieuw Archi. Wisk.*, 3(18):61-65, 1970.

[Car75] L. Carlitz. Generating functions for a special class of permutations. *Proceedings of the American Mathematical Society*, 47:251-256, 1975.

[ELR02] R. Ehrenborg, M. Levin, M. A. Readdy. A probabilistic approach to the descent statistic. *Journal of Combinatorial Theory Series A*, 98(1):150-162, 2002.

[FS09] P. Flajolet, R. Sedgewick. *Analytic combinatorics.* Cambridge University Press, Cambridge, 2009.

[LM83] D. J. Leeming, A. MacLeod. Generalized Euler number sequences: asymptotic estimates and congruences. *Canadian Journal of Mathematics*, 35:526-546, 1983.

[Luc14] J. M. Luck. On the frequencies of patterns of rises and falls. *Physica A: Statistical Mechanics and its Applications*, 407:252-275, 2014.

[Mar16] P. Marchal. Rectangular Young tableaux and the Jacobi ensemble. *Discrete Mathematics and Theoretical Computer Science Proceedings*, BC:839-850, 2016.

[MRR10] A. Mendes, J. B. Remmel, A. Riehl. Permutations with k-regular descent patterns. Permutation patterns. *London Mathematical Society Lecture Note Series,* 376:259-285, Cambridge University Press, Cambridge, 2010.

[Niv68] I. Niven. A combinatorial problem of finite sequences. *Nieuw Arch. Wiskunde*, 3(16):116-123, 1968.

[Vie79] G. Viennot. Permutations ayant une forme donnée. *Discrete Mathematics*, 26(3):279-284, 1979.