Notes on Improving the arithmetic of Kummer lines

DAMIEN ROBERT

Abstract. We explain some improvements to the arithmetic of Kummer lines.

1. Introduction

This is a summery of results that will be presented in a trilogy of articles on the arithmetic of Kummer lines.

- In [BRS23], we focus on the general theory of models of Kummer lines, the conversion between them, and the arithmetic properties of their 2-torsion point (with the relationship between the ramification, the 2-Tate pairing, the 2-theta group and their Galois representation). We also develop an hybrid arithmetic, combining the best of the (twisted) theta and Montgomery world.
- In [Rob22], we give a general framework to find equations for 2-isogenies, doubling and differential additions on a Kummer model (the formula crucially depend on the arithmetic property of the 2-torsion alluded too above). We explain how to find differential applications formula which factor through a 2-isogeny. As an application we develop a novel time/memory trade off for the Montgomery ladder.
- In [Rob23] we develop the arithmetic of the biextension associated to the divisor $2(0_E)$ on some Kummer models. We derive from this efficient pairing formula.

Part 1. Summary

In isogeny based cryptography, it is standard to work with the Montgomery model of a Kummer line. In the case where we have an extra point of 2-torsion T_2 along with the standard point of 2-torsion $T_1 = (0 : 1)$ (as happens for supersingular curves), we can use T_2 to speed up the arithmetic.

2. Hybrid arithmetic

(This is joint work with Nicolas Sarkis):

2.1. Hybrid arithmetic for scalar multiplication.

In the Montgomery ladder for computing $m.P$, we use one doubling and one mixed differential addition by step. In the Montgomery model, doubling is $2M + 2S + 1m_0$ while a mdiffAdd (where we assume our base point $P = (X_P : 1)$ is normalised) is $3M + 2S$, so a ladder step is $5M + 4S + 1m_0$. Here m_0 denotes a multiplication by a curve constant (typically the coefficient A of the Montgomery curve, or rather $(A + 2)/4$).

When T_2 is rational, we can also use a twisted theta model, where doubling is $4S + 2m_0$, and mdiffAdd is $3M+2S+1m_0$, so a ladder step is $3M+6S+3m_0$. (There is a $1M-1S-1m_0$ tradeoff where a ladder step is $4M + 5S + 2m_0$.)

Date: November 12, 2023.

The two models differ by the translation by T_2, we can exploit that to combine the best of both worlds: using an hybrid arithmetic where doubling is $4S + 2m_0$ and mdiffAdd is $3M + 2S$.

2.2. **Hybrid arithmetic for 2^n-isogenies.** In the Montgomery model, a 2-isogeny codomain costs $2S$, an image costs $4M$, and doubling cost $2S+2M+2m_0$ (because our curve coefficients are given by a projective point $(A : C)$ and we can no longer assume that $C = 1$). In practice, it is customary to use 4-isogenies instead where the codomain cost $4S$ and images $2S + 6M$.

In the twisted theta model, a 2-isogeny codomain costs $2S$, an image costs $2S + 2M$, and doubling cost $4S + 4m_0$.

Again, it is possible to use an hybrid arithmetic, with an image costing $2S + 2M$ and doubling costing $2S + 2M + 2m_0$.

This also allows to use 2-isogenies rather than 4-isogenies for the optimal strategy computation, leading to more room to find optimal parameters.

3. A time/memory trade off for scalar multiplication on Kummer lines

Although the Montgomery ladder is very efficient, for fast scalar multiplication the twisted Edward model is often faster because it allows for a time/memory trade off by using the window-NAF method to reduce the amount of additions.

However, when the scalar is a secret, these time/memory trade off are often susceptible to side channel attacks, so although signing on Curve25519 is implemented in Edwards coordinate, the DH key exchange uses the Montgomery ladder.

It might seem that a time/memory trade off is not possible on a Kummer line because standard additions are not available. In the second part of this talk, when T_2 is rational, we present a novel approach that:

(1) does a precomputation of points $P_i = (X_i : Z_i)$ costing $2S + 1m_0$ by bit, and requiring to store two field coefficients by bit.
(2) using this precomputation, the ladder then costs $2S + 1m_0$ for doubling, and $4M$ for a differential addition by bit.

The total cost, including the precomputation, is thus of $4S + 4M + 2m_0$, and further scalar multiples with the same base point then cost $2S + 4M + 1m_0$. Here it does not matter whether $P = (X_P : Z_P)$ is normalised or not.

We stress that the scalar multiplication still uses a ladder approach, with one doubling and one differential addition by bit, thus retaining the same side channel resistance as the standard Montgomery ladder. (We recall that the Montgomery ladder without precomputation costs $5M + 4S + 1m_0$ when the base point P is normalised, and $6M + 4S + 1m_0$ if P is not normalised.)

If we know that P will be used several time (like for the first step of a DH key exchange), we can increase the precomputation to normalise the points $P_i = (X_i/Z_i : 1)$. This costs one field division by bit, and reduces the storage to one field coefficient by bit.

The multiples $m.P$ then cost $2S+3M+1m_0$ by bit, significantly improving on the standard ladder.

Unfortunately, for Curve25519 the point T_2 is not rational. But its 2-isogeneous curve is a Montgomery curve with full rational 2-torsion, so by computing an isogeny at the beginning and the end we can still use our novel time/memory trade off on Curve25519 (however, unlike Curve25519, the curve constant on the isogeneous curve is not small, so we don't gain as much as if we had selected from the beginning a suitable curve with a small m_0).

4. Pairings on Kummer lines

For pairing based cryptography on elliptic curves, it is convenient to use the Tate pairing with $P \in \mathbb{G}_1 \subset E(\mathbb{F}_q)$, $Q \in \mathbb{G}_2 \subset E(\mathbb{F}_{q^k})$, and k even to allow for denominator elimination.

Counting only operations involving the big field $\mathbb{F}_{q^k}$, Miller's algorithm cost $1M+1S+1m$ by doubling, and $1M+1m$ by addition. Here $1m$ denotes a multiplication between a coefficient in $\mathbb{F}_q$ and a coefficient in $\mathbb{F}_{q^k}$.

When denominator elimination is not possible (because k is odd or Q is not in $\mathbb{G}_2$), the cost becomes $2M + 2S + 1m$ by doubling, and $2M + 1m$ by addition.

For isogeny based cryptography however, we cannot assume that one of our point is in a small field. The generic cost of the Tate pairing then becomes $5S + 15M$ for doublings, and $4S + 20M$ for additions; much more expansive than a simple scalar multiplication.

In the third part of this talk, we work out the arithmetic of the biextension associated to the divisor $2(0_E)$ on the Montgomery model of a Kummer line with full rational 2-torsion. We derive from this an efficient ladder like algorithm for pairings computation.

When P is in the small field $E(\mathbb{F}_q)$ and Q is in the big field $E(\mathbb{F}_{q^k})$, our Tate pairing algorithm costs (counting only operations in the big field) $2S + 1M + 2m$ by bits. This is competitive with the standard Miller's algorithm, except when denominator elimination is available.

For a generic pairing computation where both points are in the same field, our ladder algorithm costs $7S + 9M$ by bit, which is closer to the cost of a scalar multiplication via the Montgomery ladder.

We also explain how to compute a standard exponentiation (rather than a ladder) in the biextension, this allows to use window-NAF methods. Our algorithm costs $5S + 6M$ for a doubling, and $6S + 24M$ for an addition[1] This suggests that the second algorithm will be faster than the first one when using a window $w \geq 5$ (or when computing pairings between points of 2^n-torsion).

Part 2. Formulae

5. Models

We focus on the arithmetic of the Kummer line E of a Montgomery model with full rational 2-torsion. We represent the point of 2-torsion which is not $T_1 = (0 : 1)$ by $T_2 = (A^2 : B^2)$; and the other one is $T_2' = (B^2 : A^2)$.

Translating by T_2 be get the $\theta tw'$ (aka θ'^2) model. This allows to combine the arithmetic of the Montgomery model and (twisted) theta model.

Also the quotient $E' = E/T_1$ is also a Montgomery model with full rational 2-torsion, so we can exploit the symmetry between E and E' in our arithmetic by factorising through the isogeny $f : E \to E'$. Here when we mention a Montgomery model, we assume we have full rational 2-torsion, except if we explicitly say so.

If E has a theta model $(a : b)$, then the two torsion is $T_1 = (-a : b), T_2 = (b : a), T_2' = (-b : a)$, and $R_1 = (1 : 0), R_1' = (0 : 1)$ are 4-torsion points above $T_1 = (-a : b)$, and $R_2 = (1 : 1), R_2' = (1 : -1)$ above $T_2 = (b : a)$. Here $(A^2 : B^2) = (a^2 + b^2 : a^2 - b^2)$.

In the $\theta tw'$ model, the neutral point becomes $0 = (A^2 : B^2)$, the 2-torsion $T_1 = (B^2 : A^2), T_2 = (1 : 0), T_2' = (0 : 1)$, and the 4-torsion is $R_1 = (1 : 1), R_1' = (-1 : 1)$, and $R_2 = (a' : b') = (a + b : a - b), R_2' = (b', a')$.

[1]Standard additions are not available on a Kummer line, but we can compute them over the biextension!

In particular, the 4-torsion point $R_2 = (a' : b')$ above $T_2 = (1 : 0)$ allows to recover $(a : b)$. This model has the same ramification as the Montgomery model, except the neutral point is $(A^2 : B^2)$ which would be a point of 2-torsion T_2 on the Montgomery model, hence why they differ by translation by T_2: $(x : z) \mapsto (A^2x - B^2z : B^2x - A^2z)$.

In the Montgomery model, the neutral point is $0 = (1 : 0)$, the 2-torsion is $T_1 = (0 : 1)$, $T_2 = (A^2 : B^2)$, $T_2' = (B^2 : A^2)$, and the four torsion is $R_1 = (-1 : 1), R_1 = (1 : 1)$, $R_2 = (b' : a'), R_2' = (b' : a')$.

In the $\theta tw'$ model, the isogeny f with kernel by T_1 is given by $(x : z) \mapsto ((x+z)^2/a^2 : (x-z)^2/b^2)$. The neutral point of E' is then $(a^2 : b^2)$, T_2, T_2' are mapped to $(b^2 : a^2)$, R_1 is mapped to $(1 : 0)$, R_1' to $(0 : 1)$, R_2, R_2' to $(1 : 1)$. The dual isogeny $\tilde{f}$ is given by $(x : z) \mapsto ((x+z)^2/A^2 : (x-z)^2/B^2)$.

6. Arithmetic

In the Montgomery model, doubling is $2M + 2S + 1m_0$ while a diffAdd is $4M + 2SS$, a mdiffAdd $3M + 2S$, so a ladder step is $5M + 4S + 1m_0$.

In the (twisted) theta model, doubling is $4S + 2m_0$, diffAdd is $4M + 2S + 1m_0$ and mdiffAdd is $3M + 2S + 1m_0$, so a ladder step is $3M + 6S + 3m_0$. There is a $1M - 1S - 1m_0$ tradeoff where a ladder step is $4M + 5S + 2m_0$

These assume that our starting point P is normalised, else add $1M$ by bit.

The doubling $P \mapsto 2.P$ in twisted theta can be interpreted as $P \mapsto 2.P + T_2$ in the Montgomery model. Keeping track of the translation by T_2 we then have a hybrid ladder which cost $3M + 6S + 2m_0$.

Likewise, if we do a differential addition with points in twisted theta / Montgomery with the formula from the other model, then using $P, Q, P - Q + T_2$ will give $P + Q + T_2$, and using $P, Q + T_2, P - Q$ will give $P + Q$.

7. Isogenies

In the Montgomery model, for a 2-isogeny computation the codomain cost $2S$, and image cost $4M$, while doubling cost $4M + 2S$. (We cannot assume our constants are normalised like we are on the same curve because we keep switching curve). A 4-isogeny can be computed in $4S$ for the codomain, and $6M + 2S$ for images.

In the (twisted) theta model, the codomain cost $2S$ and image cost $2M + 2S$. But a doubling is $4M + 4S$.

The twisted theta image can be interpreted as $P \mapsto f(P) + T_2'$ in the Montgomery model, which can thus be computed in $2M + 2S$. Since T_2' is in the kernel of the next isogeny, this does not affect the next image (until the very last step).

Concretely, in the $\theta tw'$ model the isogeny with kernel T_2 is given by $g : (x : z) \mapsto (((x+z)/a + (x-z)/b)^2 : ((x+z)/a - (x-z)/b)^2)$. We recall that $(a : b)$ can be recovered from R_2. The neutral point is then $g(0) = g(T_2) = (a'^2 : b'^2), g(T_1) = g(T_2') = (b'^2 : a'^2)$, $g(R_1) = g(R_1') = (1 : 1), g(R_2) = (1 : 0), g(R_2') = (0 : 1)$.

8. Time-Memory trade off for the arithmetic

In the theta model, the arithmetic ladder stems from the duplication formula: $\theta_E(P + Q) \star \theta_E(P - Q) = H(\theta'_{E'}(f(P)) \star \theta'_{E'}(f(Q)))$.

The ladder use two steps for the differential addition (doubling is a special case where $P - Q = 0$): compute $f(P)$ via $\theta_E(P) \star \theta_E(P) = H(\theta'_{E'}(f(P)) \star \theta'_{E'}(f(0)))$. This costs $2S + 1m_0$. Do the same for $f(Q)$. Then use $\theta_E(P + Q) \star \theta_E(P - Q) = H(\theta'_{E'}(f(P)) \star$

$\theta'_{E'}(f(Q)))$ to compute $(P+Q)\star(P-Q)$ in $2M$, and then $P+Q$ in again $2M$ (or $1M$ if $P-Q$ is normalised).

A large part of the ladder is hence spent in isogeny images. Let $f_1=f$, $f_2=\tilde{f}\circ f_1$, $f_3=f\circ f_2$, $f_4=\tilde{f}\circ f_3$ and so on. Assume we had $f_{i+1}(nP), f_{i+1}((n+1))P$. Then from the duplication formula, we could directly find $f_i(2nP), f_i(2(n+1)P), f_i((2n+1)P)$.

The doublings only require the points $f_i(0_E)$ which are given by the two curves E and E'. However the differential addition needs $f_i(P)$. So what we can do is compute $f_i(P), f_i(0_E)$ then apply our duplication formula. This inverse the order: rather than doing two isogeny images and two duplication at each step, we compute all the images first and then do all the duplications. We gain because the images $f_i(0_E)$ are free. We could expect to gain $2S+1m_0$, but because our points $f_i(P)$ are no longer normalised, we only gain $2S+1m_0-M$ compared to the normal ladder with a normalised P.

In summary: we do a precomputation phase with all the $f_i(P)$. This cost $2S+1m_0$ by bit, along with 2 field coefficients. Then we do our duplication formula: this cost $2S+1m_0$ for our doublings, and $4M$ for our differential additions (again, because the $f_i(P)$ are not normalised). The final cost including the precomputation is $4M+4S+2m_0$. Further multiplication with the same base point P will cost $4M+2S+1m_0$. We note that this cost is the same whether P is normalised or not (because even if P is normalised, the $f_i(P)$ won't be).

When we know in advance P will be used (for public key encryption, or the first phase of DH key exchange), it is worth it to normalise the $f_i(P)$ at the cost of $1I$ by bit (the storage is then 1 coeff by bit). Then scalar multiplication will cost $3M+2S+1m_0$.

The big advantage compared to other time/memory trade off with elliptic curves (naf, window, …) is that the scalar multiplication is still a ladder with a double and diff add by bit, hence much less susceptible to side channel attack.

The same principle apply to the twisted theta model $\theta tw'$, but we need some careful translation by $f_i(T_2)$: for the differential addition we assume that we have $f_{i+1}(nP), f_{i+1}((n+1)P+T_2)$ (say) and we compute $f_i((2n+1)P+T_2)$. (Doublings are no problem). We obtain the same cost as in the θ model, except the initial translation by the two torsion point; likewise in the Montgomery cap Legendre model.

The formula are as follow (pending typos): given $(x_Pi : z_Pi)$, the isogenous point P_{i+1} is given by: $X=(x_Pi^2+z_Pi^2)*b_i^2 : (x_Pi^2-z_Pi^2)*a_i^2)$. From P_{i+1} we can compute $2P_i$ via the dual isogeny: $(X+Z)^2*b_{i+1}, (X-Z)^2*a_{i+1})$. The more interesting part is the differential addition, given $P_{i+1}=(xgP : zgP), Q_{i+1}+T_{2_{i+1}}=(xgQ' : zgQ'), (P-Q)_i=(xPQ : zPQ)$ we recover $(P+Q)_i$ via: $s=(xgP+zgP)(xgQ'+zgQ'); t=(xgP-zgP)(xgQ'-zgQ'); u=s+t; v=s-t; X=u/(xPQ+zPQ); Z=v/(xPQ-zPQ); (P+Q)_i=(X+Z : X-Z)$.

For Curve25519, since the two torsion is not rational, we need to move via a 2-isogeny to the curve above it which is both Montgomery and has full rational two torsion. Unfortunately the constant is large, so the cost of $4M+4S+2m_0$ when including the precomputation is essentially the same as with a standard Montgomery ladder: $5M+4S+1m_0$ (assuming P is normalised; we gain $1M$ on a non normalised point). Still, with the normalised precomputation, the cost of $3M+2S+1m_0$ is still very interesting, even with a large m_0.

The reason we work to work on the Montgomery cap Legendre model, is that if we want the relations $x(P+Q)z(P+Q), x(P-Q)z(P-Q)$ to factor through the isogeny f with kernel a 2-torsion point T, we need T to be of Montgomery type (equivalently the Tate pairing $e(T,T)=1$, or the symmetric element in the theta group above T is rational). So the curve needs to be Montgomery, but the isogeneous curve should be too (because we go back and forth between the two curves), which is equivalent to the starting curve being in Legendre form.

9. Pairings

On a Kummer line, it is useful to interpret pairings as coming from the biextension law [Gro72; Sta08] associated to the divisor $2(0_E)$. It is shown in [Gro72] how the biextension gives rise to the Weil pairing, and [Sta08] extends this to the Tate pairing.

I can reinterpret the biextension law as follow: the key point is that with a symmetric line bundle, there is a *canonical* isomorphism $t_P^* L \otimes t_Q^* L \otimes t_R^* L \otimes t_S^* L \simeq t_U^* L \otimes t_V^* L \otimes t_W^* L \otimes t_X^* L$ whenever $P + Q + R + S = 2Z, U = Z - P, V = Z - Q, W = Z - R, X = Z - S$.

Specialising, we get partial group law on trivialisation of line bundle: $\tilde{0}, \widetilde{P}, \widetilde{Q}, \widetilde{P - Q} \mapsto \widetilde{P + Q}, \tilde{0}, \widetilde{P}, \widetilde{Q}, \widetilde{R}, \widetilde{P + Q}, \widetilde{P + R}, \widetilde{Q + R} \mapsto P + \widetilde{Q} + R$.

(Note: in [Sta08] the biextension appears in the guise of elliptic nets. From our point of view, we can reinterpret elliptic nets as trivialisation of the line bundle $D = (0_E)$ at points P, notably by specifying the value of $Z(P)$ where Z is the section of (0_E). A slight difficulty is that Z has a zero on 0_E, so we need some offset to compute the pairings. The remarkable thing about elliptic nets is that even through we are on level 1 we can still compute the arithmetic of biextension through the linear recurrence of elliptic nets, see [Sta08] for details.

In [LR10; LR15], the biextension is hidden through the guise of the analytic Riemann relations giving the transcendental group law.)

We represent an element $g_{P,Q}$ of the biextension by the trivialisations $\tilde{x}, \widetilde{x + P}, \widetilde{x + Q}, x + \widetilde{P} + Q$. Changing the trivialisations by $\lambda_x, \lambda_P, \lambda_Q, \lambda_{P+Q}$ give the same element iff $\lambda_x \lambda_{P+Q} = \lambda_P \lambda_Q$. The Tate pairing is then given by $g_{P,Q}^{\ell}$, which can be computed from $\ell\widetilde{P}, \ell\widetilde{P + Q}$, which in turn can be computed from a three way Montgomery ladder: 1 doubling and 2 differential addition by step.

In the theta or twisted theta model, using [LR10; LR15] this amount to $7S + 7M + 2m_0$ by bit, assuming our base points are normalised (else add $2M$ by bit). These extend to the Montgomery model with rational two torsion (simply translate at the beginning and end to go to the $\theta tw'$ model). For a generic model unfortunately the standard formula for doubling and diff add are not the ones giving the biextension group law, we are off by some constant.

By comparison, generic pairing computations in the Jacobian model cost $15M + 5S$ for doubling, and $20M + 4S$ by addition.

We can also do a standard exponentiation on $g_{P,Q}$ on our biextension, this allows from the standard NAF and windowing method. We can do additions on the biextension model (at least with our representation), even through we are on the Kummer line on the underlying curve!

I worked out the formula in the theta model, using [LR15; LR16]: doubling cost 1 double and 1 diff add on the underlying curve, for a cost of $4M + 5S + 2m_0$. Addition is more complicated: on the underlying curve this amount to one (projective) compatible addition which cost $27M$ (I am not distinguishing M, S and m_0 here), followed by an affine three way addition which cost $17M$, for a grand total of $44M$. But since our base points are always the same (the ones we computed for our window), we can do some precomputations for these steps, and the compatible addition then cost $17M$, and the three way addition $13M$, for a total of $30M$.

Since doubling is $11M$, this might be competitive with the ladder method (which costs $16M$ by bit) when using a NAF-window with $w \geq 5$.

By contrast, the generic cost of the Tate pairing using Miller's standard algorithm is $5S + 15M$ for doublings, and $4S + 20M$ for additions.

9.1. **The Tate pairing for pairing based cryptogrpahy.** For pairing based cryptography on elliptic curves, it is convenient to use the Tate pairing with $P \in_1 \subset E(\mathbb{F}_q)$, $Q \in_2 \subset E(\mathbb{F}_{q^k})$, and k even to allow for denominator elimination.

Counting only operations involving the big field $\mathbb{F}_{q^k}$, Miller's algorithm cost $1M+1S+1m$ by doubling, and $1M+1m$ by addition. Here $1m$ denotes a multiplication between a coefficient in $\mathbb{F}_q$ and a coefficient in $\mathbb{F}_{q^k}$.

When denominator elimination is not possible (because k is odd or Q is not in $_2$), the cost becomes $2M + 2S + 1m$ by doubling, and $2M + 1m$ by addition.

Using our arithmetic of biextension on Kummer lines, only counting the operations on the big field, we have $2S + 1M + 2m$ by bit. So better than Miller's algorithm, except when denominator elimination is available.

References

[BRS23] R. Barbulescu, D. Robert, and N. Sarkis. "Models of Kummer lines and Galois representations". June 2023. In preparation. (Cit. on p. 1).

[Gro72] A. Grothendieck. *Groupes de Monodromie en Géométrie Algébrique: SGA 7*. Springer-Verlag, 1972 (cit. on p. 6).

[LR10] D. Lubicz and D. Robert. "Efficient pairing computation with theta functions". In: ed. by G. Hanrot, F. Morain, and E. Thomé. Vol. 6197. Lecture Notes in Comput. Sci. 9th International Symposium, Nancy, France, ANTS-IX, July 19-23, 2010, Proceedings. Springer–Verlag, July 2010. DOI: 10.1007/978-3-642-14518-6_21. URL: http://www.normalesup.org/~robert/pro/publications/articles/pairings.pdf. Slides: 2010-07-ANTS-Nancy.pdf (30min, International Algorithmic Number Theory Symposium (ANTS-IX), July 2010, Nancy), HAL: hal-00528944. (Cit. on p. 6).

[LR15] D. Lubicz and D. Robert. "A generalisation of Miller's algorithm and applications to pairing computations on abelian varieties". In: *Journal of Symbolic Computation* 67 (Mar. 2015), pp. 68–92. DOI: 10.1016/j.jsc.2014.08.001. URL: http://www.normalesup.org/~robert/pro/publications/articles/optimal.pdf. HAL: hal-00806923, eprint: 2013/192. (Cit. on p. 6).

[LR16] D. Lubicz and D. Robert. "Arithmetic on Abelian and Kummer Varieties". In: *Finite Fields and Their Applications* 39 (May 2016), pp. 130–158. DOI: 10.1016/j.ffa.2016.01.009. URL: http://www.normalesup.org/~robert/pro/publications/articles/arithmetic.pdf. HAL: hal-01057467, eprint: 2014/493. (Cit. on p. 6).

[Rob22] D. Robert. "Arithmetic on Kummer lines". Oct. 2022. In preparation. (Cit. on p. 1).

[Rob23] D. Robert. "Pairings on Kummer lines". Aug. 2023. In preparation. (Cit. on p. 1).

[Sta08] K. Stange. "Elliptic nets and elliptic curves". PhD thesis. Brown University, 2008. URL: https://repository.library.brown.edu/studio/item/bdr:309/PDF/ (cit. on p. 6).

INRIA Bordeaux–Sud-Ouest, 200 avenue de la Vieille Tour, 33405 Talence Cedex FRANCE
Email address: damien.robert@inria.fr
URL: http://www.normalesup.org/~robert/

Institut de Mathématiques de Bordeaux, 351 cours de la liberation, 33405 Talence cedex FRANCE