Montgomery ladders already compute pairings

Alessandro Sferlazza
joint work with: G. Pope, K. Reijnders, D. Robert, B. Smith

Technical University of Munich

29 April 2025,
SQIparty Workshop, Lleida

Main character: pairings of elliptic curves

Pairings are bilinear maps from subgroups/quotients of elliptic curves with nice extra properties

$$\begin{array}{rrcl} e_\ell\colon & G_1 \times G_2 & \to & \mu_\ell \subseteq k^\times \\ & (P,Q) & \mapsto & e_\ell(P,Q) \end{array} \qquad \ell \in \mathbb{N}$$

Main character: pairings of elliptic curves

Pairings are bilinear maps from subgroups/quotients of elliptic curves with nice extra properties

$$\begin{array}{rccc} e_\ell : & G_1 \times G_2 & \to & \mu_\ell \subseteq k^\times \\ & (P, Q) & \mapsto & e_\ell(P, Q) \end{array} \qquad \ell \in \mathbb{N}$$

Countless uses in crypto:

- curve-based and pairing-based cryptography

Main character: pairings of elliptic curves

Pairings are bilinear maps from subgroups/quotients of elliptic curves with nice extra properties

$$\begin{array}{rccl} e_\ell \colon & G_1 \times G_2 & \to & \mu_\ell \subseteq k^\times \\ & (P,Q) & \mapsto & e_\ell(P,Q) \end{array} \qquad \ell \in \mathbb{N}$$

Countless uses in crypto:

- curve-based and pairing-based cryptography $\rightsquigarrow$ highly optimized parameters:
  - field characteristic $p = \operatorname{char} k$ with fast arithmetic
  - P, Q on a fixed curve E with small/nice coefficients

Main character: pairings of elliptic curves

Pairings are bilinear maps from subgroups/quotients of elliptic curves with nice extra properties

$$e_\ell : \begin{array}{rcl} G_1 \times G_2 & \to & \mu_\ell \subseteq k^\times \\ (P, Q) & \mapsto & e_\ell(P, Q) \end{array} \qquad \ell \in \mathbb{N}$$

Countless uses in crypto:

- curve-based and pairing-based cryptography $\rightsquigarrow$ highly optimized parameters:
  - field characteristic $p = \operatorname{char} k$ with fast arithmetic
  - P, Q on a fixed curve E with small/nice coefficients
- isogeny-based crypto:
  - parameters p, E already constrained (e.g. for rational torsion)

Main character: pairings of elliptic curves

Pairings are bilinear maps from subgroups/quotients of elliptic curves with nice extra properties

$$\begin{array}{rccc} e_\ell\colon & G_1 \times G_2 & \to & \mu_\ell \subseteq k^\times \\ & (P,Q) & \mapsto & e_\ell(P,Q) \end{array} \qquad \ell \in \mathbb{N}$$

Countless uses in crypto:

- curve-based and pairing-based cryptography $\rightsquigarrow$ highly optimized parameters:
  - field characteristic $p = \operatorname{char} k$ with fast arithmetic
  - P, Q on a fixed curve E with small/nice coefficients
- isogeny-based crypto:
  - parameters p, E already constrained (e.g. for rational torsion)

$\rightsquigarrow$ need fast generic pairing.

Main character: pairings of elliptic curves

Pairings are bilinear maps from subgroups/quotients of elliptic curves with nice extra properties

$$e_\ell : \begin{array}{ccc} G_1 \times G_2 & \to & \mu_\ell \subseteq k^\times \\ (P,Q) & \mapsto & e_\ell(P,Q) \end{array} \qquad \ell \in \mathbb{N}$$

Countless uses in crypto:

- curve-based and pairing-based cryptography $\rightsquigarrow$ highly optimized parameters:
  - field characteristic $p = \operatorname{char} k$ with fast arithmetic
  - P, Q on a fixed curve E with small/nice coefficients
- isogeny-based crypto:
  - parameters p, E already constrained (e.g. for rational torsion)

$\rightsquigarrow$ need fast generic pairing.

Cost of generic pairings per bit of ℓ:

	Tate pairing	Weil pairing
State of the art [CLZ24][1]	11.3M + 7.7S + 20.7A	2 · Tate pairing
[Rob24][2] $\rightsquigarrow$ our work	9M + 6S + 16A	

[1] Cai, Lin, Zhao, *Pairing Optimizations for Isogeny-based Cryptosystems*, eprint.iacr.org/2024/575

[2] Robert, *Fast pairings via biextensions and cubical arithmetic*, eprint.iacr.org/2024/517

Montgomery ladder

Computes scalar multiplication $P \mapsto [\ell]P$
using x-only arithmetic: $P = (X_P : Z_P)$

Montgomery ladder

Computes scalar multiplication $P \mapsto [\ell]P$ using x-only arithmetic: $P = (X_P : Z_P)$

Forgetting about Y, sign ambiguity $\pm P \rightsquigarrow$ can't add $P + Q$ with the usual group law.

On $E/\pm$ we have two operations

$$\mathrm{xDBL}\colon P \mapsto [2]P$$

$$\mathrm{xADD}\colon (P_1, P_2; P_1 - P_2) \mapsto P_1 + P_2$$

Montgomery ladder

Computes scalar multiplication $P \mapsto [\ell]P$ using x-only arithmetic: $P = (X_P : Z_P)$

Forgetting about Y, sign ambiguity $\pm P \rightsquigarrow$ can't add $P + Q$ with the usual group law.

On $E/\pm$ we have two operations

$$\mathrm{xDBL}\colon P \mapsto [2]P$$

$$\mathrm{xADD}\colon (P_1, P_2; P_1 - P_2) \mapsto P_1 + P_2$$

Combine them into a

$$\textsc{Ladder}\colon (\ell, P) \mapsto ([\ell]P, [\ell+1]P).$$

$[\ell]P$	$[\ell+1]P$
$\dots$	$\dots$
$[2n]P$	$[2n+1]P$
$[n]P$	$[n+1]P$
$\dots$	$\dots$
P	$2P$
0_E	P

Montgomery ladder

Computes scalar multiplication $P \mapsto [\ell]P$ using x-only arithmetic: $P = (X_P : Z_P)$

Forgetting about Y, sign ambiguity $\pm P \rightsquigarrow$ can't add $P + Q$ with the usual group law.

On $E/\pm$ we have two operations

$$\mathrm{xDBL}\colon P \mapsto [2]P$$
$$\mathrm{xADD}\colon (P_1, P_2; P_1 - P_2) \mapsto P_1 + P_2$$

Combine them into a

$$\textsc{Ladder}\colon (\ell, P) \mapsto ([\ell]P, [\ell+1]P).$$

0_E P

Montgomery ladder

Computes scalar multiplication $P \mapsto [\ell]P$
using x-only arithmetic: $P = (X_P : Z_P)$

Forgetting about Y, sign ambiguity $\pm P \rightsquigarrow$
can't add $P + Q$ with the usual group law.

On $E/\pm$ we have two operations

$$\mathrm{xDBL}\colon P \mapsto [2]P$$
$$\mathrm{xADD}\colon (P_1, P_2; P_1 - P_2) \mapsto P_1 + P_2$$

Combine them into a

$$\textsc{Ladder}\colon (\ell, P) \mapsto ([\ell]P, [\ell+1]P).$$

P $2P$

0_E P

Montgomery ladder

Computes scalar multiplication $P \mapsto [\ell]P$ using x-only arithmetic: $P = (X_P : Z_P)$

Forgetting about Y, sign ambiguity $\pm P \rightsquigarrow$ can't add $P + Q$ with the usual group law.

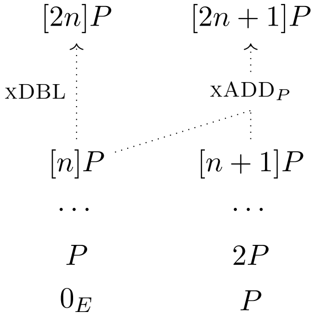
On $E/\pm$ we have two operations

$$\mathrm{xDBL}\colon P \mapsto [2]P$$

$$\mathrm{xADD}\colon (P_1, P_2; P_1 - P_2) \mapsto P_1 + P_2$$

Combine them into a

$$\textsc{Ladder}\colon (\ell, P) \mapsto ([\ell]P, [\ell+1]P).$$



Montgomery ladder

Computes scalar multiplication $P \mapsto [\ell]P$ using x-only arithmetic: $P = (X_P : Z_P)$

Forgetting about Y, sign ambiguity $\pm P \rightsquigarrow$ can't add $P + Q$ with the usual group law.

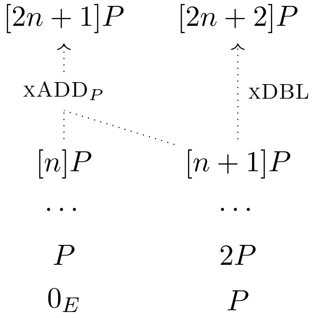
On $E/\pm$ we have two operations

$$\text{xDBL}\colon P \mapsto [2]P$$

$$\text{xADD}\colon (P_1, P_2; P_1 - P_2) \mapsto P_1 + P_2$$

Combine them into a

$$\textsc{Ladder}\colon (\ell, P) \mapsto ([\ell]P, [\ell+1]P).$$



Montgomery ladder

Computes scalar multiplication $P \mapsto [\ell]P$ using x-only arithmetic: $P = (X_P : Z_P)$

Forgetting about Y, sign ambiguity $\pm P \rightsquigarrow$ can't add $P + Q$ with the usual group law.

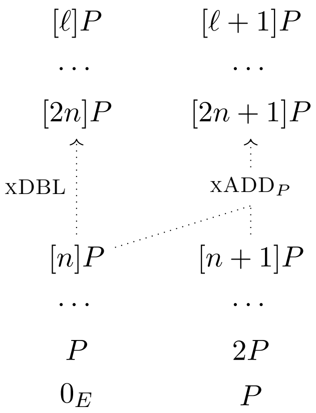
On $E/\pm$ we have two operations

$$\mathrm{xDBL}\colon P \mapsto [2]P$$

$$\mathrm{xADD}\colon (P_1, P_2; P_1 - P_2) \mapsto P_1 + P_2$$

Combine them into a

$$\textsc{Ladder}\colon (\ell, P) \mapsto ([\ell]P, [\ell+1]P).$$



Montgomery ladder

Computes scalar multiplication $P \mapsto [\ell]P$ using x-only arithmetic: $P = (X_P : Z_P)$

Forgetting about Y, sign ambiguity $\pm P \rightsquigarrow$ can't add $P + Q$ with the usual group law.

On $E/\pm$ we have two operations

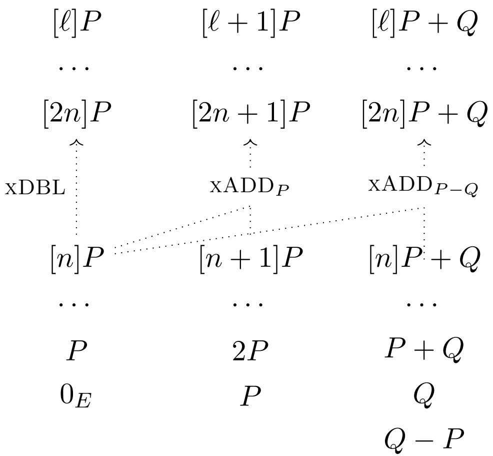
$$\mathrm{xDBL}\colon P \mapsto [2]P$$

$$\mathrm{xADD}\colon (P_1, P_2; P_1 - P_2) \mapsto P_1 + P_2$$

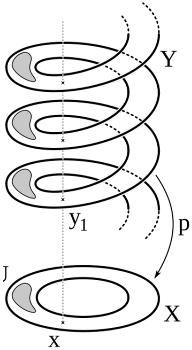
Combine them into a

$$\textsc{Ladder}\colon (\ell, P) \mapsto ([\ell]P, [\ell+1]P).$$

Generalizable to a $\textsc{3PtLadder}$ with offset Q.
Need input $\pm(P - Q)$.

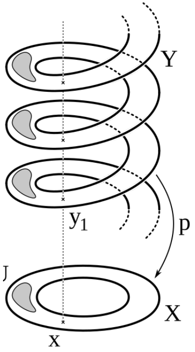


The role of monodromy



The role of monodromy

Torsion relation in $E(k)$		Torsion relation in $\mathrm{Pic}^0(E)(k)$		Monodromy in $\mathrm{Div}^0(E)$
$[\ell]P = 0$	$\leftrightarrow$	$\left[\ell(P) - \ell(0_E)\right] = 0$	$\leftrightarrow$	$\ell(P) - \ell(0_E) = \mathrm{div}\, f_{\ell,P}$



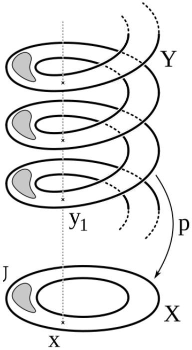
The role of monodromy

Torsion relation in $E(k)$ $\leftrightarrow$ Torsion relation in $\mathrm{Pic}^0(E)(k)$ $\leftrightarrow$ Monodromy in $\mathrm{Div}^0(E)$

$$[\ell]P = 0 \quad \leftrightarrow \quad \big[\ell(P) - \ell(0_E)\big] = 0 \quad \leftrightarrow \quad \ell(P) - \ell(0_E) = \mathrm{div}\, f_{\ell,P}$$

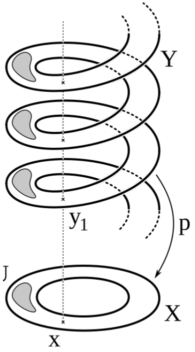
The non-reduced Tate pairing of degree $\ell \in \mathbb{N}$ over k stems from monodromy:

$$\begin{aligned} e_{T,\ell}\colon \quad E[\ell](k) \times E(k)/[\ell]E(k) &\to k^\times/(k^\times)^\ell \\ (P, [Q]) &\mapsto f_{\ell,P}(Q) \end{aligned}$$



The role of monodromy

Torsion relation in $E(k)$		Torsion relation in $\mathrm{Pic}^0(E)(k)$		Monodromy in $\mathrm{Div}^0(E)$
$[\ell]P = 0$	$\leftrightarrow$	$\left[\ell(P) - \ell(0_E)\right] = 0$	$\leftrightarrow$	$\ell(P) - \ell(0_E) = \mathrm{div}\, f_{\ell,P}$



The non-reduced Tate pairing of degree $\ell \in \mathbb{N}$ over k stems from monodromy:

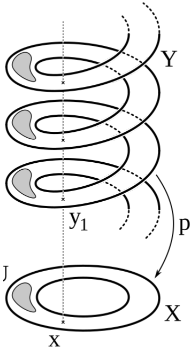
$$\begin{aligned} e_{T,\ell}\colon \quad E[\ell](k) \times E(k)/[\ell]E(k) &\to k^\times/(k^\times)^\ell \\ (P, [Q]) &\mapsto f_{\ell,P}(Q) \end{aligned}$$

Miller's algorithm:

- Compute $P \mapsto [\ell]P = 0$, say using an addition chain $P, 2P, ..., \ell P$

The role of monodromy

Torsion relation in $E(k)$		Torsion relation in $\mathrm{Pic}^0(E)(k)$		Monodromy in $\mathrm{Div}^0(E)$
$[\ell]P = 0$	$\leftrightarrow$	$\left[\ell(P) - \ell(0_E)\right] = 0$	$\leftrightarrow$	$\ell(P) - \ell(0_E) = \mathrm{div}\, f_{\ell,P}$



The non-reduced Tate pairing of degree $\ell \in \mathbb{N}$ over k stems from monodromy:

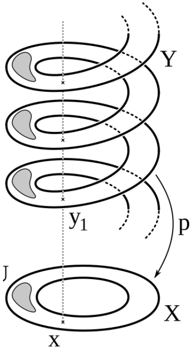
$$\begin{array}{rccc} e_{T,\ell}\colon & E[\ell](k) \times E(k)/[\ell]E(k) & \to & k^\times/(k^\times)^\ell \\ & (P, [Q]) & \mapsto & f_{\ell,P}(Q) \end{array}$$

Miller's algorithm:

- Compute $P \mapsto [\ell]P = 0$, say using an addition chain $P, 2P, \ldots, \ell P$
- Accumulate evaluated line functions: $f_{\ell,P}(Q) = \prod_j \frac{l_{[n_j]P,[m_j]P}(Q)}{l_{[m_j]P,[m_j]P}(Q)}$

The role of monodromy

Torsion relation in $E(k)$		Torsion relation in $\mathrm{Pic}^0(E)(k)$		Monodromy in $\mathrm{Div}^0(E)$
$[\ell]P = 0$	$\leftrightarrow$	$\left[\ell(P) - \ell(0_E)\right] = 0$	$\leftrightarrow$	$\ell(P) - \ell(0_E) = \mathrm{div}\, f_{\ell,P}$



The non-reduced Tate pairing of degree $\ell \in \mathbb{N}$ over k stems from monodromy:

$$\begin{array}{rccc} e_{T,\ell}\colon & E[\ell](k) \times E(k)/[\ell]E(k) & \to & k^\times/(k^\times)^\ell \\ & (P,[Q]) & \mapsto & f_{\ell,P}(Q) \end{array}$$

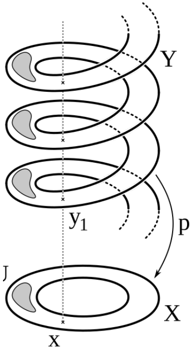
Miller's algorithm:

- Compute $P \mapsto [\ell]P = 0$, say using an addition chain $P, 2P, ..., \ell P$
- Accumulate evaluated line functions: $f_{\ell,P}(Q) = \prod_j \frac{l_{[n_j]P,[m_j]P}(Q)}{l_{[m_j]P,[m_j]P}(Q)}$

$\rightsquigarrow$ pairing from the intermediate additions!

The role of monodromy

Torsion relation in $E(k)$		Torsion relation in $\mathrm{Pic}^0(E)(k)$		Monodromy in $\mathrm{Div}^0(E)$
$[\ell]P = 0$	$\leftrightarrow$	$\left[\ell(P) - \ell(0_E)\right] = 0$	$\leftrightarrow$	$\ell(P) - \ell(0_E) = \operatorname{div} f_{\ell,P}$



The non-reduced Tate pairing of degree $\ell \in \mathbb{N}$ over k stems from monodromy:

$$e_{T,\ell}\colon \quad E[\ell](k) \times E(k)/[\ell]E(k) \to k^\times/(k^\times)^\ell, \quad (P, [Q]) \mapsto f_{\ell,P}(Q)$$

Miller's algorithm:

- Compute $P \mapsto [\ell]P = 0$, say using an addition chain $P, 2P, \ldots, \ell P$
- Accumulate evaluated line functions: $f_{\ell,P}(Q) = \prod_j \frac{l_{[n_j]P,[m_j]P}(Q)}{l_{[m_j]P,[m_j]P}(Q)}$

  $\rightsquigarrow$ pairing from the intermediate additions!

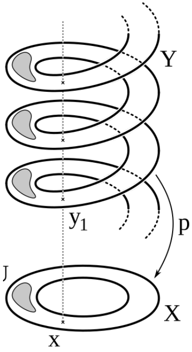
Monodromy already appears in the Montgomery ladder alone:

- Start with $0_E = (1:0)$ and $P = (X_P : Z_P)$
- Perform $\textsc{Ladder}(P, \ell)$: get $[\ell]P = (X_{\ell P} : 0) = (1:0)$

  $\rightsquigarrow$ $X_{\ell P}$ is a monodromy factor.

The role of monodromy

Torsion relation in $E(k)$		Torsion relation in $\mathrm{Pic}^0(E)(k)$		Monodromy in $\mathrm{Div}^0(E)$
$[\ell]P = 0$	$\leftrightarrow$	$\left[\ell(P) - \ell(0_E)\right] = 0$	$\leftrightarrow$	$\ell(P) - \ell(0_E) = \mathrm{div}\, f_{\ell,P}$



The non-reduced Tate pairing of degree $\ell \in \mathbb{N}$ over k stems from monodromy:

$$\begin{array}{rccc} e_{T,\ell}\colon & E[\ell](k) \times E(k)/[\ell]E(k) & \to & k^\times/(k^\times)^\ell \\ & (P,[Q]) & \mapsto & f_{\ell,P}(Q) \end{array}$$

Miller's algorithm:

- Compute $P \mapsto [\ell]P = 0$, say using an addition chain $P, 2P, \ldots, \ell P$
- Accumulate evaluated line functions: $f_{\ell,P}(Q) = \prod_j \frac{l_{[n_j]P,[m_j]P}(Q)}{l_{[m_j]P,[m_j]P}(Q)}$

  $\rightsquigarrow$ pairing from the intermediate additions!

Monodromy already appears in the Montgomery ladder alone:

- Start with $0_E = (1:0)$ and $P = (X_P : Z_P)$
- Perform $\textsc{Ladder}(P, \ell)$: get $[\ell]P = (X_{\ell P} : 0) = (1:0)$

  $\rightsquigarrow$ $X_{\ell P}$ is a monodromy factor. Projective coordinates carry meaning!

Montgomery ladders *almost* compute pairings

$$P = (x_P : 1) \in E[\ell], \quad Q = (x_Q : 1), \quad P - Q = (x_{P-Q} : 1)$$

Montgomery ladders *almost* compute pairings

$$P = (x_P : 1) \in E[\ell], \quad Q = (x_Q : 1), \quad P - Q = (x_{P-Q} : 1)$$

Look at monodromy factors using ladders:

$$\begin{array}{l} 0_E = (1,0) \\ Q = (x_Q, 1) \end{array} \xrightarrow{\textsc{3PtLadder}(\ell, P, Q; P-Q)} \begin{array}{ll} [\ell]P = (X_{\ell P}, 0) & \text{differ by } \lambda_P = X_{\ell P} \\ [\ell]P + Q = (X_{\ell P+Q}, Z_{\ell P+Q}) & \text{differ by } \lambda_Q = Z_{\ell P+Q} \end{array}$$

Montgomery ladders *almost* compute pairings

$$P = (x_P : 1) \in E[\ell], \quad Q = (x_Q : 1), \quad P - Q = (x_{P-Q} : 1)$$

Look at monodromy factors using ladders:

$$\begin{matrix} 0_E = (1,0) \\ Q = (x_Q, 1) \end{matrix} \xrightarrow{\textsc{3PtLadder}(\ell, P, Q; P-Q)} \begin{matrix} [\ell]P = (X_{\ell P}, 0) & \text{differ by } \lambda_P = X_{\ell P} \\ [\ell]P + Q = (X_{\ell P+Q}, Z_{\ell P+Q}) & \text{differ by } \lambda_Q = Z_{\ell P+Q} \end{matrix}$$

From this we get the Tate pairing!

$$\lambda_Q / \lambda_P = e_{T,\ell}(P, Q)^2 \cdot \text{STUFF}$$

Montgomery ladders *almost* compute pairings

$$P = (x_P : 1) \in E[\ell], \quad Q = (x_Q : 1), \quad P - Q = (x_{P-Q} : 1)$$

Look at monodromy factors using ladders:

$$\begin{matrix} 0_E = (1, 0) \\ Q = (x_Q, 1) \end{matrix} \xrightarrow{\textsc{3PtLadder}(\ell, P, Q; P-Q)} \begin{matrix} [\ell]P = (X_{\ell P}, 0) & \text{differ by } \lambda_P = X_{\ell P} \\ [\ell]P + Q = (X_{\ell P+Q}, Z_{\ell P+Q}) & \text{differ by } \lambda_Q = Z_{\ell P+Q} \end{matrix}$$

From this we get the Tate pairing! squared, + garbage

$$\lambda_Q / \lambda_P = e_{T,\ell}(P, Q)^2 \cdot \text{STUFF}$$

Montgomery ladders *almost* compute pairings

$$P = (x_P : 1) \in E[\ell], \quad Q = (x_Q : 1), \quad P - Q = (x_{P-Q} : 1)$$

Look at monodromy factors using ladders:

$$\begin{matrix} 0_E = (1,0) \\ Q = (x_Q, 1) \end{matrix} \xrightarrow{\textsc{3PtLadder}(\ell,P,Q;P-Q)} \begin{matrix} [\ell]P = (X_{\ell P}, 0) & \text{differ by } \lambda_P = X_{\ell P} \\ [\ell]P + Q = (X_{\ell P+Q}, Z_{\ell P+Q}) & \text{differ by } \lambda_Q = Z_{\ell P+Q} \end{matrix}$$

From this we get the Tate pairing! squared, + garbage

$$\lambda_Q / \lambda_P = e_{T,\ell}(P, Q)^2 \cdot \text{STUFF}$$

More precisely, $\text{STUFF} = \dfrac{(4x_P)^{\ell\cdot(\neg\ell+1)}}{(4x_P)^{\ell\cdot\neg\ell}(4x_Q)^{\ell}(4x_{P-Q})^{\neg\ell}}$ depends on[3]

- initial input coordinates
- bit representation of ℓ.

[3]notation: $\neg\ell$ = bitwise negation of the bit representation of ℓ

Montgomery ladders *almost* compute pairings

$$P = (x_P : 1) \in E[\ell], \quad Q = (x_Q : 1), \quad P - Q = (x_{P-Q} : 1)$$

Look at monodromy factors using ladders:

$$\begin{matrix} 0_E = (1,0) \\ Q = (x_Q, 1) \end{matrix} \xrightarrow{\textsc{3PtLadder}(\ell,P,Q;P-Q)} \begin{matrix} [\ell]P = (X_{\ell P}, 0) & \text{differ by } \lambda_P = X_{\ell P} \\ [\ell]P + Q = (X_{\ell P+Q}, Z_{\ell P+Q}) & \text{differ by } \lambda_Q = Z_{\ell P+Q} \end{matrix}$$

From this we get the Tate pairing! squared, + garbage

$$\lambda_Q/\lambda_P = e_{T,\ell}(P,Q)^2 \cdot \textsc{stuff}$$

More precisely, $\textsc{stuff} = \dfrac{(4x_P)^{\ell\cdot(\neg\ell+1)}}{(4x_P)^{\ell\cdot\neg\ell}(4x_Q)^{\ell}(4x_{P-Q})^{\neg\ell}}$ depends on[3]

- initial input coordinates
- bit representation of ℓ.

Solution: compute $\textsc{stuff}$ and divide it out...

or better: edit the $\textsc{Ladder}$ to get rid of $\textsc{stuff}$.

[3]notation: $\neg\ell$ = bitwise negation of the bit representation of ℓ

Montgomery ladders compute pairings

Consider $\mathrm{xADD}(P, Q; P - Q) = (X_{P+Q}, Z_{P+Q})$.

Montgomery ladders compute pairings

Consider $\text{xADD}(P, Q; P - Q) = (X_{P+Q}, Z_{P+Q})$.

Modify into cADD: different projective scaling of the output (X_{P+Q}, Z_{P+Q})

$$\begin{aligned} X_{P+Q} &= Z_{P-Q}\,(T+U)^2\,, \\ Z_{P+Q} &= X_{P-Q}\,(T-U)^2\,. \end{aligned} \quad \rightsquigarrow \quad \begin{aligned} X_{P+Q} &= (4X_{P-Q})^{-1} \cdot (T+U)^2\,, \\ Z_{P+Q} &= (4Z_{P-Q})^{-1} \cdot (T-U)^2\,. \end{aligned}$$

Montgomery ladders compute pairings

Consider $\quad \mathrm{xADD}(P, Q; P-Q) = (X_{P+Q}, Z_{P+Q})$.
Modify into cADD: different projective scaling of the output (X_{P+Q}, Z_{P+Q})

$$\begin{aligned} X_{P+Q} &= Z_{P-Q}\,(T+U)^2\,, \\ Z_{P+Q} &= X_{P-Q}\,(T-U)^2\,. \end{aligned} \quad \rightsquigarrow \quad \begin{aligned} X_{P+Q} &= (4X_{P-Q})^{-1} \cdot (T+U)^2\,, \\ Z_{P+Q} &= (4Z_{P-Q})^{-1} \cdot (T-U)^2\,. \end{aligned}$$

Montgomery arithmetic using xDBL and the new cADD: cubical arithmetic.

Montgomery ladders compute pairings

Consider $\text{xADD}(P, Q; P - Q) = (X_{P+Q}, Z_{P+Q})$.

Modify into cADD: different projective scaling of the output (X_{P+Q}, Z_{P+Q})

$$\begin{aligned} X_{P+Q} &= Z_{P-Q}\,(T+U)^2\,, \\ Z_{P+Q} &= X_{P-Q}\,(T-U)^2\,. \end{aligned} \quad \rightsquigarrow \quad \begin{aligned} X_{P+Q} &= (4X_{P-Q})^{-1} \cdot (T+U)^2\,, \\ Z_{P+Q} &= (4Z_{P-Q})^{-1} \cdot (T-U)^2\,. \end{aligned}$$

Montgomery arithmetic using xDBL and the new cADD: cubical arithmetic.

Replace now cADD into the ladder.

Then $\text{cLadder}(\ell, P, Q; P - Q) \mapsto (\ell P, \ell P + Q)$ in (X, Z)-coordinates:

$$\lambda'_Q / \lambda'_P = X_{\ell P} / Z_{\ell P + Q} = e_{T,\ell}(P, Q)^2 \qquad \text{without extra STUFF!}$$

Montgomery ladders compute pairings

Consider $\quad \mathrm{xADD}(P, Q; P-Q) = (X_{P+Q}, Z_{P+Q})$.

Modify into cADD: different projective scaling of the output (X_{P+Q}, Z_{P+Q})

$$\begin{aligned} X_{P+Q} &= Z_{P-Q}\,(T+U)^2\,, \\ Z_{P+Q} &= X_{P-Q}\,(T-U)^2\,. \end{aligned} \quad \rightsquigarrow \quad \begin{aligned} X_{P+Q} &= (4X_{P-Q})^{-1}\cdot(T+U)^2\,, \\ Z_{P+Q} &= (4Z_{P-Q})^{-1}\cdot(T-U)^2\,. \end{aligned}$$

Montgomery arithmetic using xDBL and the new cADD: cubical arithmetic.

Replace now cADD into the ladder.

Then $\quad \mathrm{cLadder}(\ell, P, Q; P-Q) \mapsto (\ell P, \ell P + Q) \quad$ in (X, Z)-coordinates:

$$\lambda'_Q/\lambda'_P = X_{\ell P}/Z_{\ell P+Q} = e_{T,\ell}(P,Q)^2 \qquad \text{without extra STUFF!}$$

- The square is not a problem when ℓ is odd ✓

Montgomery ladders compute pairings

Consider $\text{xADD}(P, Q; P - Q) = (X_{P+Q}, Z_{P+Q})$.
Modify into cADD: different projective scaling of the output (X_{P+Q}, Z_{P+Q})

$$\begin{aligned} X_{P+Q} &= Z_{P-Q}\,(T+U)^2\,, \\ Z_{P+Q} &= X_{P-Q}\,(T-U)^2\,. \end{aligned} \quad \rightsquigarrow \quad \begin{aligned} X_{P+Q} &= (4X_{P-Q})^{-1}\cdot(T+U)^2\,, \\ Z_{P+Q} &= (4Z_{P-Q})^{-1}\cdot(T-U)^2\,. \end{aligned}$$

Montgomery arithmetic using xDBL and the new cADD: cubical arithmetic.

Replace now cADD into the ladder.
Then $\text{cLadder}(\ell, P, Q; P - Q) \mapsto (\ell P, \ell P + Q)$ in (X, Z)-coordinates:

$$\lambda'_Q/\lambda'_P = X_{\ell P}/Z_{\ell P+Q} = e_{T,\ell}(P,Q)^2 \qquad \text{without extra STUFF!}$$

- The square is not a problem when ℓ is odd ✓ $\quad$ ℓ even $\longrightarrow$ small trick to avoid the square

Montgomery ladders compute pairings

Consider $\text{xADD}(P, Q; P-Q) = (X_{P+Q}, Z_{P+Q})$.

Modify into cADD: different projective scaling of the output (X_{P+Q}, Z_{P+Q})

$$\begin{aligned} X_{P+Q} &= Z_{P-Q}\,(T+U)^2\,, \\ Z_{P+Q} &= X_{P-Q}\,(T-U)^2\,. \end{aligned} \quad \rightsquigarrow \quad \begin{aligned} X_{P+Q} &= (4X_{P-Q})^{-1}\cdot(T+U)^2\,, \\ Z_{P+Q} &= (4Z_{P-Q})^{-1}\cdot(T-U)^2\,. \end{aligned}$$

Montgomery arithmetic using xDBL and the new cADD: cubical arithmetic.

Replace now cADD into the ladder.

Then $\text{cLadder}(\ell, P, Q; P-Q) \mapsto (\ell P, \ell P + Q)$ in (X, Z)-coordinates:

$$\lambda'_Q/\lambda'_P = X_{\ell P}/Z_{\ell P+Q} = e_{T,\ell}(P,Q)^2 \qquad \text{without extra STUFF!}$$

- The square is not a problem when ℓ is odd ✓ $\quad$ ℓ even $\longrightarrow$ small trick to avoid the square
- Just minor tweak needed in the conversion $\text{xADD} \longrightarrow \text{cADD}$
  $\rightsquigarrow$ easy optimized, constant-time implementation.[4]

[4] Rust and Sagemath libraries provided at https://github.com/GiacomoPope/cubical-pairings

Montgomery ladders compute pairings

Consider $\text{xADD}(P, Q; P - Q) = (X_{P+Q}, Z_{P+Q})$.

Modify into cADD: different projective scaling of the output (X_{P+Q}, Z_{P+Q})

$$\begin{aligned} X_{P+Q} &= Z_{P-Q}\,(T+U)^2\,, \\ Z_{P+Q} &= X_{P-Q}\,(T-U)^2\,. \end{aligned} \quad \rightsquigarrow \quad \begin{aligned} X_{P+Q} &= (4X_{P-Q})^{-1}\cdot(T+U)^2\,, \\ Z_{P+Q} &= (4Z_{P-Q})^{-1}\cdot(T-U)^2\,. \end{aligned}$$

Montgomery arithmetic using xDBL and the new cADD: cubical arithmetic.

Replace now cADD into the ladder.

Then $\text{cLadder}(\ell, P, Q; P - Q) \mapsto (\ell P, \ell P + Q)$ in (X, Z)-coordinates:

$$\lambda'_Q/\lambda'_P = X_{\ell P}/Z_{\ell P+Q} = e_{T,\ell}(P,Q)^2 \qquad \text{without extra STUFF!}$$

- The square is not a problem when ℓ is odd ✓ $\quad$ ℓ even $\longrightarrow$ small trick to avoid the square
- Just minor tweak needed in the conversion $\text{xADD} \longrightarrow \text{cADD}$
  $\rightsquigarrow$ easy optimized, constant-time implementation.[4]
- Inverses can be pre-computed and batched: only one inversion per pairing

[4] Rust and Sagemath libraries provided at `https://github.com/GiacomoPope/cubical-pairings`

Cubical arithmetic

It seems: there's a preferred projective scaling in the output of xADD. Not a coincidence!

Cubical arithmetic

It seems: there's a preferred projective scaling in the output of xADD. Not a coincidence!

Algebraic statement:

- Projective coordinates X, Z are *global sections* of a line bundle $\mathcal{L}$

Cubical arithmetic

It seems: there's a preferred projective scaling in the output of xADD. Not a coincidence!

Algebraic statement:

- Projective coordinates X, Z are *global sections* of a line bundle $\mathcal{L}$
- There is a canonical isomorphism of line bundles

$$t_{P_1}^*\mathcal{L} \otimes t_{P_2}^*\mathcal{L} \otimes t_{P_3}^*\mathcal{L} \otimes t_{P_1+P_2+P_3}^*\mathcal{L} \cong t_{P_2+P_3}^*\mathcal{L} \otimes t_{P_1+P_3}^*\mathcal{L} \otimes t_{P_1+P_2}^*\mathcal{L} \otimes \mathcal{L}$$

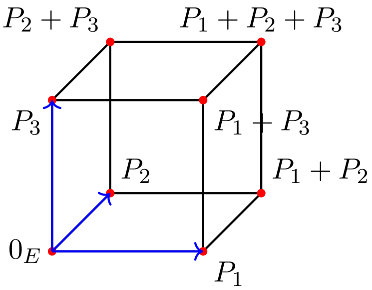
Cubical arithmetic

It seems: there's a preferred projective scaling in the output of xADD. Not a coincidence!

Algebraic statement:

- Projective coordinates X, Z are *global sections* of a line bundle $\mathcal{L}$
- There is a canonical isomorphism of line bundles

$$t_{P_1}^*\mathcal{L} \otimes t_{P_2}^*\mathcal{L} \otimes t_{P_3}^*\mathcal{L} \otimes t_{P_1+P_2+P_3}^*\mathcal{L} \cong t_{P_2+P_3}^*\mathcal{L} \otimes t_{P_1+P_3}^*\mathcal{L} \otimes t_{P_1+P_2}^*\mathcal{L} \otimes \mathcal{L}$$



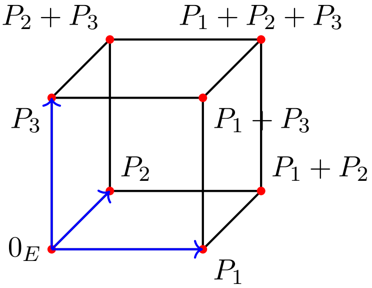
Cubical arithmetic

It seems: there's a preferred projective scaling in the output of $\textsc{xADD}$. Not a coincidence!

Algebraic statement:

- Projective coordinates X, Z are *global sections* of a line bundle $\mathcal{L}$
- There is a canonical isomorphism of line bundles

$$t_{P_1}^*\mathcal{L} \otimes t_{P_2}^*\mathcal{L} \otimes t_{P_3}^*\mathcal{L} \otimes t_{P_1+P_2+P_3}^*\mathcal{L} \cong t_{P_2+P_3}^*\mathcal{L} \otimes t_{P_1+P_3}^*\mathcal{L} \otimes t_{P_1+P_2}^*\mathcal{L} \otimes \mathcal{L}$$



Read $t_P^*\mathcal{L}$ as: choose scaling of coordinates X_P, Z_P

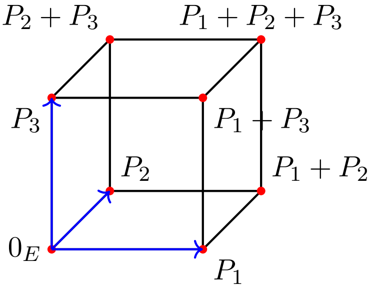
Cubical arithmetic

It seems: there's a preferred projective scaling in the output of xADD. Not a coincidence!

Algebraic statement:

- Projective coordinates X, Z are *global sections* of a line bundle $\mathcal{L}$
- There is a canonical isomorphism of line bundles

$$t^*_{P_1}\mathcal{L} \otimes t^*_{P_2}\mathcal{L} \otimes t^*_{P_3}\mathcal{L} \otimes t^*_{P_1+P_2+P_3}\mathcal{L} \cong t^*_{P_2+P_3}\mathcal{L} \otimes t^*_{P_1+P_3}\mathcal{L} \otimes t^*_{P_1+P_2}\mathcal{L} \otimes \mathcal{L}$$



Read $t^*_P\mathcal{L}$ as: choose scaling of coordinates X_P, Z_P

Given coordinates of 7 vertices,

isomorphism above $\Longrightarrow$ canonical choice for the 8th

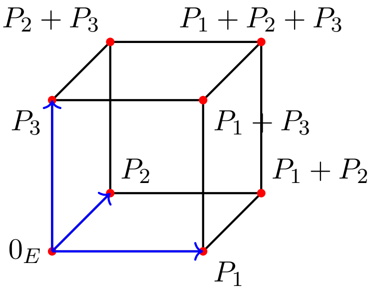
Cubical arithmetic

It seems: there's a preferred projective scaling in the output of xADD. Not a coincidence!

Algebraic statement:

- Projective coordinates X, Z are *global sections* of a line bundle $\mathcal{L}$
- There is a canonical isomorphism of line bundles

$$t_{P_1}^*\mathcal{L} \otimes t_{P_2}^*\mathcal{L} \otimes t_{P_3}^*\mathcal{L} \otimes t_{P_1+P_2+P_3}^*\mathcal{L} \cong t_{P_2+P_3}^*\mathcal{L} \otimes t_{P_1+P_3}^*\mathcal{L} \otimes t_{P_1+P_2}^*\mathcal{L} \otimes \mathcal{L}$$



Read $t_P^*\mathcal{L}$ as: choose scaling of coordinates X_P, Z_P

Given coordinates of 7 vertices,
isomorphism above $\Longrightarrow$ canonical choice for the 8th

We get cADD (and cDBL) as special case:
Let $(P_1, P_2, P_3) = (P, Q, -Q)$. The vertices

$$(P,\ Q,\ -Q,\ P,\ 0,\ P+Q,\ P-Q,\ 0)$$

Fixing $P, Q, P-Q$ we get $P+Q$ uniquely!

Cubical arithmetic as a way to get Miller functions

Consider the projective coordinates X, Z.

Cubical arithmetic as a way to get Miller functions

Consider the projective coordinates X, Z.

Warning: they're global sections of a line bundle, *not* rational functions in $k(E)$.
The rational (meromorphic) functions corresp. to X, Z are $x, 1$ respectively.

Cubical arithmetic as a way to get Miller functions

Consider the projective coordinates X, Z.

Warning: they're global sections of a line bundle, *not* rational functions in $k(E)$.
The rational (meromorphic) functions corresp. to X, Z are $x, 1$ respectively.

However, see $0_E = (1 : 0)$. The coordinate Z has a zero at 0_E (with multiplicity!)
Global sections have a zero locus. There is a reasonable notion of divisor of zeroes:

$$\mathrm{div}_0(Z) = 2(0_E), \qquad \mathrm{div}_0(Z(\cdot + P)) = 2(-P).$$

Cubical arithmetic as a way to get Miller functions

Consider the projective coordinates X, Z.

Warning: they're global sections of a line bundle, *not* rational functions in $k(E)$.
The rational (meromorphic) functions corresp. to X, Z are $x, 1$ respectively.

However, see $0_E = (1:0)$. The coordinate Z has a zero at 0_E (with multiplicity!)
Global sections have a zero locus. There is a reasonable notion of divisor of zeroes:

$$\mathrm{div}_0(Z) = 2(0_E), \qquad \mathrm{div}_0(Z(\cdot + P)) = 2(-P).$$

Idea: compute some ratio $g(\cdot) = \dfrac{Z(\cdot + P_1)\cdots Z(\cdot + P_m)}{Z(\cdot + Q_1)\cdots Z(\cdot + Q_m)}$ (in gen. not a function in $k(E)$)

where the P_i, Q_j are all compatible via the cubical arithmetic.

Cubical arithmetic as a way to get Miller functions

Consider the projective coordinates X, Z.

Warning: they're global sections of a line bundle, *not* rational functions in $k(E)$.
The rational (meromorphic) functions corresp. to X, Z are $x, 1$ respectively.

However, see $0_E = (1:0)$. The coordinate Z has a zero at 0_E (with multiplicity!)
Global sections have a zero locus. There is a reasonable notion of divisor of zeroes:

$$\operatorname{div}_0(Z) = 2(0_E), \qquad \operatorname{div}_0(Z(\cdot + P)) = 2(-P).$$

Idea: compute some ratio $g(\cdot) = \dfrac{Z(\cdot + P_1)\cdots Z(\cdot + P_m)}{Z(\cdot + Q_1)\cdots Z(\cdot + Q_m)}$ (in gen. not a function in $k(E)$)

where the P_i, Q_j are all compatible via the cubical arithmetic.

When the P_i, Q_j are chosen carefully, we can get a rational function $g \in k(E)$, satisfying

$$\operatorname{div} g = 2(-P_1) + \cdots + 2(-P_m) - 2(-Q_1) - \cdots - 2(-Q_m)$$

Cubical arithmetic as a way to get Miller functions

Consider the projective coordinates X, Z.

Warning: they're global sections of a line bundle, *not* rational functions in $k(E)$.
The rational (meromorphic) functions corresp. to X, Z are $x, 1$ respectively.

However, see $0_E = (1:0)$. The coordinate Z has a zero at 0_E (with multiplicity!)
Global sections have a zero locus. There is a reasonable notion of divisor of zeroes:

$$\mathrm{div}_0(Z) = 2(0_E), \qquad \mathrm{div}_0(Z(\cdot + P)) = 2(-P).$$

Idea: compute some ratio $g(\cdot) = \dfrac{Z(\cdot + P_1)\cdots Z(\cdot + P_m)}{Z(\cdot + Q_1)\cdots Z(\cdot + Q_m)}$ (in gen. not a function in $k(E)$)

where the P_i, Q_j are all compatible via the cubical arithmetic.

When the P_i, Q_j are chosen carefully, we can get a rational function $g \in k(E)$, satisfying

$$\mathrm{div}\, g = 2(-P_1) + \cdots + 2(-P_m) - 2(-Q_1) - \cdots - 2(-Q_m)$$

Miller fns: $P \in E[\ell]$. Then $f_{\ell,P} : R \mapsto \dfrac{Z(R + \ell P) Z(R)^{\ell - 1}}{Z(P)^{\ell}}$ has divisor $2\big(\ell(0) - \ell(-P)\big)$

End of the theory!

Some applications now

Application: multi-dimensional discrete logarithms

- Consider a torsion basis $\langle P, Q \rangle = E[N]$, with N smooth.
- Let $R \in E[N]$.

Application: multi-dimensional discrete logarithms

- Consider a torsion basis $\langle P, Q \rangle = E[N]$, with N smooth.
- Let $R \in E[N]$. DLog problem: recover (a, b) s.t. $R = [a]P + [b]Q$.

Application: multi-dimensional discrete logarithms

- Consider a torsion basis $\langle P, Q \rangle = E[N]$, with N smooth.
- Let $R \in E[N]$. DLog problem: recover (a, b) s.t. $R = [a]P + [b]Q$.

How to solve? Weil pairing: $e_N : E[N] \times E[N] \to \mu_N$.

Application: multi-dimensional discrete logarithms

- Consider a torsion basis $\langle P, Q\rangle = E[N]$, with N smooth.
- Let $R \in E[N]$. DLog problem: recover (a, b) s.t. $R = [a]P + [b]Q$.

How to solve? Weil pairing: $e_N\colon E[N] \times E[N] \to \mu_N$.

- Alternating: $e(P, P) = 1$
- Non-degenerate: if P has order N, there is Q s.t. $e(P, Q)$ has order N.
  $e(P, Q)$ has order $N \iff \langle P, Q\rangle = E[N]$

Application: multi-dimensional discrete logarithms

- Consider a torsion basis $\langle P, Q \rangle = E[N]$, with N smooth.
- Let $R \in E[N]$. DLog problem: recover (a, b) s.t. $R = [a]P + [b]Q$.

How to solve? Weil pairing: $e_N \colon E[N] \times E[N] \to \mu_N$.

- Alternating: $e(P, P) = 1$
- Non-degenerate: if P has order N, there is Q s.t. $e(P, Q)$ has order N.
  $e(P, Q)$ has order $N \iff \langle P, Q \rangle = E[N]$

In many isogeny applications, use Tate pairing: similar properties, faster to compute.

Application: multi-dimensional discrete logarithms

- Consider a torsion basis $\langle P, Q\rangle = E[N]$, with N smooth.
- Let $R \in E[N]$. DLog problem: recover (a, b) s.t. $R = [a]P + [b]Q$.

How to solve? Weil pairing: $e_N : E[N] \times E[N] \to \mu_N$.

- Alternating: $e(P, P) = 1$
- Non-degenerate: if P has order N, there is Q s.t. $e(P, Q)$ has order N.
  $e(P, Q)$ has order $N \iff \langle P, Q\rangle = E[N]$

In many isogeny applications, use Tate pairing: similar properties, faster to compute.

Some details:

$$\zeta_0 = e_N(P, Q) \qquad \text{has order } N$$

Application: multi-dimensional discrete logarithms

- Consider a torsion basis $\langle P, Q\rangle = E[N]$, with N smooth.
- Let $R \in E[N]$. DLog problem: recover (a, b) s.t. $R = [a]P + [b]Q$.

How to solve? Weil pairing: $e_N : E[N] \times E[N] \to \mu_N$.

- Alternating: $e(P, P) = 1$
- Non-degenerate: if P has order N, there is Q s.t. $e(P, Q)$ has order N.
  $e(P, Q)$ has order $N \iff \langle P, Q\rangle = E[N]$

In many isogeny applications, use Tate pairing: similar properties, faster to compute.

Some details:

$$\zeta_0 = e_N(P, Q) \qquad \text{has order } N$$

$$h_b = e_N(R, P) = e_N([a]P + [b]Q, P) = \zeta_0^{-b}$$

Application: multi-dimensional discrete logarithms

- Consider a torsion basis $\langle P, Q\rangle = E[N]$, with N smooth.
- Let $R \in E[N]$. DLog problem: recover (a, b) s.t. $R = [a]P + [b]Q$.

How to solve? Weil pairing: $e_N : E[N] \times E[N] \to \mu_N$.

- Alternating: $e(P, P) = 1$
- Non-degenerate: if P has order N, there is Q s.t. $e(P, Q)$ has order N.
  $e(P, Q)$ has order $N \iff \langle P, Q\rangle = E[N]$

In many isogeny applications, use Tate pairing: similar properties, faster to compute.

Some details:

$$\zeta_0 = e_N(P, Q) \qquad \text{has order } N$$

$$h_b = e_N(R, P) = e_N([a]P + [b]Q, P) = \zeta_0^{-b}$$

$$h_a = e_N(R, Q) = e_N([a]P + [b]Q, Q) = \zeta_0^{a}$$

Application: multi-dimensional discrete logarithms

- Consider a torsion basis $\langle P, Q\rangle = E[N]$, with N smooth.
- Let $R \in E[N]$. DLog problem: recover (a, b) s.t. $R = [a]P + [b]Q$.

How to solve? Weil pairing: $e_N : E[N] \times E[N] \to \mu_N$.

- Alternating: $e(P, P) = 1$
- Non-degenerate: if P has order N, there is Q s.t. $e(P, Q)$ has order N.
  $e(P, Q)$ has order $N \iff \langle P, Q\rangle = E[N]$

In many isogeny applications, use Tate pairing: similar properties, faster to compute.

Some details:

$$\zeta_0 = e_N(P, Q) \qquad \text{has order } N$$
$$h_b = e_N(R, P) = e_N([a]P + [b]Q, P) = \zeta_0^{-b}$$
$$h_a = e_N(R, Q) = e_N([a]P + [b]Q, Q) = \zeta_0^{a}$$

$\rightsquigarrow$ DLog in μ_N, much easier

Application: multi-dimensional discrete logarithms

- Consider a torsion basis $\langle P, Q\rangle = E[N]$, with N smooth.
- Let $R \in E[N]$. DLog problem: recover (a, b) s.t. $R = [a]P + [b]Q$.

How to solve? Weil pairing: $e_N \colon E[N] \times E[N] \to \mu_N$.

- Alternating: $e(P, P) = 1$
- Non-degenerate: if P has order N, there is Q s.t. $e(P, Q)$ has order N.
  $e(P, Q)$ has order $N \iff \langle P, Q\rangle = E[N]$

In many isogeny applications, use Tate pairing: similar properties, faster to compute.

Some details:

$$\zeta_0 = e_N(P, Q) \quad \text{has order } N$$
$$h_b = e_N(R, P) = e_N([a]P + [b]Q, P) = \zeta_0^{-b}$$
$$h_a = e_N(R, Q) = e_N([a]P + [b]Q, Q) = \zeta_0^{a}$$

$\rightsquigarrow$ DLog in μ_N, much easier

Use examples: Point compression in SQIsignHD: $\sim$ 40% cost reduction
Decryption in (Q)FESTA, HD protocols...

Further applications: torsion bases, supersingularity testing

Weil pairing: $e_{W,N} \colon E[N] \times E[N] \to \mu_N$.

- Non-degenerate $\implies e(P,Q)$ has order N iff P, Q are a torsion basis.

Further applications: torsion bases, supersingularity testing

Weil pairing: $e_{W,N} \colon E[N] \times E[N] \to \mu_N$.

- Non-degenerate $\implies e(P,Q)$ has order N iff P, Q are a torsion basis.

Application #1: Torsion basis generation for very composite $N = \prod_i \ell_i$

- Sample random points P, Q
- Test they form a torsion basis by testing the order of $e(P,Q) \in \mu_N$.

Further applications: torsion bases, supersingularity testing

Weil pairing: $e_{W,N}\colon E[N] \times E[N] \to \mu_N$.

- Non-degenerate $\implies e(P,Q)$ has order N iff P,Q are a torsion basis.

Application #1: Torsion basis generation for very composite $N = \prod_i \ell_i$

- Sample random points P,Q
- Test they form a torsion basis by testing the order of $e(P,Q) \in \mu_N$.
  - $\rightsquigarrow$ Order testing in μ_N: much faster than trial multiplication $P \mapsto [N/\ell_i]P$ ✓

Further applications: torsion bases, supersingularity testing

Weil pairing: $e_{W,N} \colon E[N] \times E[N] \to \mu_N.$

- Non-degenerate $\Longrightarrow e(P,Q)$ has order N iff P,Q are a torsion basis.

Application #1: Torsion basis generation for very composite $N = \prod_i \ell_i$

- Sample random points P,Q
- Test they form a torsion basis by testing the order of $e(P,Q) \in \mu_N$.
  - $\rightsquigarrow$ Order testing in μ_N: much faster than trial multiplication $P \mapsto [N/\ell_i]P$ ✓

Application #2: Supersingularity verification

- Let $E/\mathbb{F}_{p^2}$ be a supersingular curve, say $E(\mathbb{F}_{p^2}) \cong (\mathbb{Z}/(p \pm 1)\mathbb{Z})^2$.

Further applications: torsion bases, supersingularity testing

Weil pairing: $e_{W,N} \colon E[N] \times E[N] \to \mu_N$.

- Non-degenerate $\implies e(P, Q)$ has order N iff P, Q are a torsion basis.

Application #1: Torsion basis generation for very composite $N = \prod_i \ell_i$

- Sample random points P, Q
- Test they form a torsion basis by testing the order of $e(P, Q) \in \mu_N$.
  - $\rightsquigarrow$ Order testing in μ_N: much faster than trial multiplication $P \mapsto [N/\ell_i]P$ ✓

Application #2: Supersingularity verification

- Let $E/\mathbb{F}_{p^2}$ be a supersingular curve, say $E(\mathbb{F}_{p^2}) \cong (\mathbb{Z}/(p \pm 1)\mathbb{Z})^2$.
- Try to generate a $(p+1)$-torsion basis. If SUCCESS, return "E is supersingular".
- FAIL if we find P with $[p+1]P \neq 0$. Retry few times otherwise.

Further applications: torsion bases, supersingularity testing

Weil pairing: $e_{W,N} \colon E[N] \times E[N] \to \mu_N$.

- Non-degenerate $\implies e(P,Q)$ has order N iff P,Q are a torsion basis.

Application #1: Torsion basis generation for very composite $N = \prod_i \ell_i$

- Sample random points P, Q
- Test they form a torsion basis by testing the order of $e(P,Q) \in \mu_N$.
  - $\rightsquigarrow$ Order testing in μ_N: much faster than trial multiplication $P \mapsto [N/\ell_i]P$ ✓

Application #2: Supersingularity verification

- Let $E/\mathbb{F}_{p^2}$ be a supersingular curve, say $E(\mathbb{F}_{p^2}) \cong (\mathbb{Z}/(p \pm 1)\mathbb{Z})^2$.
- Try to generate a $(p+1)$-torsion basis. If SUCCESS, return "E is supersingular".
- FAIL if we find P with $[p+1]P \neq 0$. Retry few times otherwise.
  - $\rightsquigarrow$ Probability of false negatives: 0. Probability of false positives: negligible. ✓

Further applications: torsion bases, supersingularity testing

Weil pairing: $e_{W,N}\colon E[N] \times E[N] \to \mu_N$.

- Non-degenerate $\implies e(P,Q)$ has order N iff P,Q are a torsion basis.

Application #1: Torsion basis generation for very composite $N = \prod_i \ell_i$

- Sample random points P,Q
- Test they form a torsion basis by testing the order of $e(P,Q) \in \mu_N$.
  - $\rightsquigarrow$ Order testing in μ_N: much faster than trial multiplication $P \mapsto [N/\ell_i]P$ ✓

Application #2: Supersingularity verification

- Let $E/\mathbb{F}_{p^2}$ be a supersingular curve, say $E(\mathbb{F}_{p^2}) \cong (\mathbb{Z}/(p \pm 1)\mathbb{Z})^2$.
- Try to generate a $(p+1)$-torsion basis. If SUCCESS, return "E is supersingular".
- FAIL if we find P with $[p+1]P \neq 0$. Retry few times otherwise.
  - $\rightsquigarrow$ Probability of false negatives: 0. Probability of false positives: negligible. ✓

Use-case example: CSIDH public key validation: $\sim$ 7% cost reduction.

Further directions

The theory of cubical arithmetic and biextensions applies much more generally:

- Other pairings (e.g., Ate pairing and variants):

Further directions

The theory of cubical arithmetic and biextensions applies much more generally:

- Other pairings (e.g., Ate pairing and variants):
  $\rightsquigarrow$ optimizations in pairing-based crypto, see [LRZZ25][5]

[5] Lin, Robert, Zhao, Zheng, *Biextensions in Pairing-based Cryptography*, `eprint.iacr.org/2025/670`

Further directions

The theory of cubical arithmetic and biextensions applies much more generally:

- Other pairings (e.g., Ate pairing and variants):
  $\rightsquigarrow$ optimizations in pairing-based crypto, see [LRZZ25][5]
- Other curve models: Theta, Weierstrass, Edwards, ...

[5] Lin, Robert, Zhao, Zheng, *Biextensions in Pairing-based Cryptography*, eprint.iacr.org/2025/670

Further directions

The theory of cubical arithmetic and biextensions applies much more generally:

- Other pairings (e.g., Ate pairing and variants):
  $\rightsquigarrow$ optimizations in pairing-based crypto, see [LRZZ25][5]
- Other curve models: Theta, Weierstrass, Edwards, ...
- Higher dimensions: with level-2 theta models, Weil & Tate-Lichtenbaum work similarly
  $\rightsquigarrow$ Cubical pairings already present in AVisogenies in Magma.

[5] Lin, Robert, Zhao, Zheng, *Biextensions in Pairing-based Cryptography*, `eprint.iacr.org/2025/670`

Further directions

The theory of cubical arithmetic and biextensions applies much more generally:

- Other pairings (e.g., Ate pairing and variants):
  $\rightsquigarrow$ optimizations in pairing-based crypto, see [LRZZ25][5]
- Other curve models: Theta, Weierstrass, Edwards, ...
- Higher dimensions: with level-2 theta models, Weil & Tate-Lichtenbaum work similarly
  $\rightsquigarrow$ Cubical pairings already present in AVisogenies in Magma.
  $\rightsquigarrow$ We have some initial Sagemath implementation.
  $\rightsquigarrow$ Improve it and integrate it in the Sagemath code: coding sprints!

[5] Lin, Robert, Zhao, Zheng, *Biextensions in Pairing-based Cryptography*, `eprint.iacr.org/2025/670`

Further directions

The theory of cubical arithmetic and biextensions applies much more generally:

- Other pairings (e.g., Ate pairing and variants):
  - $\rightsquigarrow$ optimizations in pairing-based crypto, see [LRZZ25][5]
- Other curve models: Theta, Weierstrass, Edwards, ...
- Higher dimensions: with level-2 theta models, Weil & Tate-Lichtenbaum work similarly
  - $\rightsquigarrow$ Cubical pairings already present in AVisogenies in Magma.
  - $\rightsquigarrow$ We have some initial Sagemath implementation.
  - $\rightsquigarrow$ Improve it and integrate it in the Sagemath code: coding sprints!

Thank you for listening! Questions?

[5] Lin, Robert, Zhao, Zheng, *Biextensions in Pairing-based Cryptography*, `eprint.iacr.org/2025/670`

Even-degree pairings

Consider an even integer $\ell = 2m$.

Even-degree pairings

Consider an even integer $\ell = 2m$.

$$P \in E[\ell](k), \quad Q \in E(k), \qquad \textsc{cLadder}(\ell, P, Q, P-Q) \mapsto \ell P,\ \ell P + Q$$

We can get the squared Tate pairing: $\lambda_P / \lambda_Q = X_{\ell P} / Z_{\ell P + Q} = e_{T,\ell}(P, Q)^2$

Even-degree pairings

Consider an even integer $\ell = 2m$.

$$P \in E[\ell](k), \quad Q \in E(k), \qquad \textsc{cLadder}(\ell, P, Q, P-Q) \mapsto \ell P,\ \ell P + Q$$

We can get the squared Tate pairing: $\quad \lambda_P/\lambda_Q = X_{\ell P}/Z_{\ell P+Q} = e_{T,\ell}(P,Q)^2$

The pairing has order dividing $\ell = 2m \rightsquigarrow$ the square loses one bit of information.

Even-degree pairings

Consider an even integer $\ell = 2m$.

$$P \in E[\ell](k), \quad Q \in E(k), \qquad \textsc{cLadder}(\ell, P, Q, P-Q) \mapsto \ell P,\ \ell P + Q$$

We can get the squared Tate pairing: $\quad \lambda_P/\lambda_Q = X_{\ell P}/Z_{\ell P+Q} = e_{T,\ell}(P,Q)^2$

The pairing has order dividing $\ell = 2m \rightsquigarrow$ the square loses one bit of information.

Step 1: only compute ladder of order $m = \ell/2$.

$$\textsc{cLadder}(m, P, Q, P-Q) \mapsto mP,\ mP + Q$$

Even-degree pairings

Consider an even integer $\ell = 2m$.

$$P \in E[\ell](k), \quad Q \in E(k), \qquad \textsc{cLadder}(\ell, P, Q, P-Q) \mapsto \ell P,\ \ell P + Q$$

We can get the squared Tate pairing: $\quad \lambda_P/\lambda_Q = X_{\ell P}/Z_{\ell P+Q} = e_{T,\ell}(P,Q)^2$

The pairing has order dividing $\ell = 2m \rightsquigarrow$ the square loses one bit of information.

Step 1: only compute ladder of order $m = \ell/2$.

$$\textsc{cLadder}(m, P, Q, P-Q) \mapsto mP,\ mP + Q$$

Step 2: *Linear translations.* $T = mP$ is a point of order 2: on the Kummer line, translation by T induces an involution. It acts linearly on coordinates, for example

$$T = (0:1). \qquad T * (X_P, Z_P) = P + T = (Z_P, X_P)$$

$$T = (A:B) \neq (0:1) \qquad T * (X_P, Z_P) = P + T = (AX_P - BZ_P, AZ_P - BX_P)$$

Even-degree pairings

Consider an even integer $\ell = 2m$.

$$P \in E[\ell](k), \quad Q \in E(k), \qquad \text{cLadder}(\ell, P, Q, P-Q) \mapsto \ell P,\ \ell P + Q$$

We can get the squared Tate pairing: $\lambda_P/\lambda_Q = X_{\ell P}/Z_{\ell P+Q} = e_{T,\ell}(P,Q)^2$

The pairing has order dividing $\ell = 2m \rightsquigarrow$ the square loses one bit of information.

Step 1: only compute ladder of order $m = \ell/2$.

$$\text{cLadder}(m, P, Q, P-Q) \mapsto mP,\ mP + Q$$

Step 2: *Linear translations.* $T = mP$ is a point of order 2: on the Kummer line, translation by T induces an involution. It acts linearly on coordinates, for example

$$T = (0:1). \qquad T * (X_P, Z_P) = P + T = (Z_P, X_P)$$

$$T = (A:B) \neq (0:1) \qquad T * (X_P, Z_P) = P + T = (AX_P - BZ_P, AZ_P - BX_P)$$

Step 3: *Monodromy.*

$mP + T$ is projectively $= 0_E$ $\rightsquigarrow$ monodromy factor λ'_P

$(mP + Q) + T$ is projectively $= Q$ $\rightsquigarrow$ monodromy factor λ'_Q

Even-degree pairings

Consider an even integer $\ell = 2m$.

$$P \in E[\ell](k), \quad Q \in E(k), \qquad \textsc{cLadder}(\ell, P, Q, P-Q) \mapsto \ell P,\ \ell P + Q$$

We can get the squared Tate pairing: $\lambda_P/\lambda_Q = X_{\ell P}/Z_{\ell P+Q} = e_{T,\ell}(P,Q)^2$

The pairing has order dividing $\ell = 2m \rightsquigarrow$ the square loses one bit of information.

Step 1: only compute ladder of order $m = \ell/2$.

$$\textsc{cLadder}(m, P, Q, P-Q) \mapsto mP,\ mP + Q$$

Step 2: *Linear translations.* $T = mP$ is a point of order 2: on the Kummer line, translation by T induces an involution. It acts linearly on coordinates, for example

$$T = (0:1). \qquad T * (X_P, Z_P) = P + T = (Z_P, X_P)$$

$$T = (A:B) \neq (0:1) \qquad T * (X_P, Z_P) = P + T = (AX_P - BZ_P, AZ_P - BX_P)$$

Step 3: *Monodromy.*

$mP + T$ is projectively $= 0_E$ $\rightsquigarrow$ monodromy factor λ'_P

$(mP + Q) + T$ is projectively $= Q$ $\rightsquigarrow$ monodromy factor λ'_Q

$$\lambda_P/\lambda_Q = X_{mP+T}/Z_{(mP+Q)+T} = e_{T,\ell}(P,Q) \qquad \text{without the square!}$$

Cubical arithmetic in different models

	cDBL	cADD
Montgomery	3M 2S	3M 2S
Theta	3M 2S	3M 3S
Weierstrass	5M 4S	8M 2S