Integer division
Newton’s iteration
\[ x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \]

Let \( f(x) = 0 \). Then

\[
|x - x_{i+1}| = \left| x - x_i + \frac{f(x_i)}{f'(x_i)} \right| \\
= \frac{|f(x_i) + f'(x_i)(x - x_i)|}{|f'(x_i)|} \\
= \frac{|f(x) - \frac{f''(\tilde{x}_i)}{2}(x - x_i)^2|}{|f'(x_i)|} \\
= \frac{1}{2} \frac{|f''(\tilde{x}_i)|}{|f'(x_i)|} |x - x_i|^2
\]

for some \( \tilde{x}_i \) between \( x \) and \( x_i \) (why?).

That is, under some assumptions, Newton’s iteration converges quadratically (which assumptions?), or, in other words, the precision is doubled at each iteration step.
Let $I(n)$ denote the number of bit operations needed for computing the first $n$ bits of the inverse of a positive integer.

**Theorem** If $M(2n) \geq 2M(n)$, then $I(n) = O(M(n))$.

**Proof** For a positive number $u$, we are going to compute the first $n$ bits of $u^{-1}$. Let

$$f(x) = 1 - ux.$$ 

Then the solution of $f(x) = 0$ is $x = \frac{1}{u}$.

Since $f'(x) = -u$, the $i$th step of Newton’s iteration is

$$x_{i+1} = x_i - \frac{1 - ux_i}{-u} = \frac{1}{u} = x_i + \frac{1}{u}(1 - ux_i)$$

$$\approx x_i + x_i(1 - ux_i) = 2x_i - ux_i^2.$$


If $ux_i = 1 - \varepsilon$, then

$$ux_{i+1} = 2ux_i - u^2x_i^2 = 2(1 - \varepsilon) - (1 - \varepsilon)^2 = 1 - \varepsilon^2.$$ 

It follows from $ux_i = 1 - \varepsilon$ and $1 \leq u$ that

$$\left| x_i - \frac{1}{u} \right| = \frac{\varepsilon}{u} < \varepsilon.$$ 

Therefore, iteration $i + 1$ doubles the number of correct bits of $x_i$. 
We may assume that $1 \leq u < 2$ (why?).

We start with $x_0 = 0.1$.

Since each iteration doubles the number of correct bits, we shall compute only $2^i$ leading bits of $x_i$, for which we need only $2^i$ leading bits of $u$.

It follows from

$$x_{i+1} = 2x_i - ux_i^2$$

that

$$I(2^{i+1}) = I(2^i) + cM(2^i) .$$

Then

$$I(n) \leq c \sum_{i=0}^{\log_2 n} M\left(\frac{n}{2^i}\right)$$

$$\leq c \sum_{i=0}^{\log_2 n} \frac{M(n)}{2^i} = 2cM(n) .$$

(why?)
Computing quotient and remainder for integers
Let $u$ and $v$ be an $n$- and an $m$-bit integers, respectively, $m \leq n$. The quotient $\lfloor u/v \rfloor$ and the remainder $\text{rem}(u, v)$ can be computed as follows.

- Compute the first $n + 1$ bits of $v^{-1}$ (how?). Denote this number by $w$.

- Then

  $$\lfloor u/v \rfloor = \lfloor vw \rfloor$$

  (why?) and

  $$\text{rem}(u, v) = u - v\lfloor u/v \rfloor$$

  (why?).

All this can be done in $O(M(n))$ bit operations (why?).
Division of polynomials
Let

\[ u(\lambda) = \sum_{i=0}^{\infty} u_i \lambda^i, \]

where \( u_0 = 1 \), and let the sequence \( x_0(\lambda), x_1(\lambda), \ldots \) of power series be defined by \( x_0(\lambda) = 1 \) and

\[ x_{i+1}(\lambda) = 2x_i(\lambda) - u(\lambda)x_i^2(\lambda), \]

\( i = 0, 1, \ldots \).

**Lemma** If \( u(\lambda)x(\lambda) \equiv 1 \mod \lambda^k \), then

\[ u(\lambda)(2x(\lambda) - u(\lambda)x^2(\lambda)) \equiv 1 \mod \lambda^{2k}. \]

Consequently, if the first \( k \) coefficients of \( x_i(\lambda) \) equal the first \( k \) coefficients of \( u^{-1}(\lambda) \), then the first \( 2k \) coefficients of \( x_{i+1}(\lambda) \) equal the first \( 2k \) coefficients of \( u^{-1}(\lambda) \).
Proof of the lemma

\[ u(\lambda)(2x(\lambda) - u(\lambda)x^2(\lambda)) = 2u(\lambda)x(\lambda) - u^2(\lambda)x^2(\lambda) \]

\[ = 1 - (1 - u(\lambda)x(\lambda))^2 \equiv 1 \mod \lambda^{2k}. \]

Since \( x_0(\lambda) = 1 \),

\[ u(\lambda)x_0(\lambda) = \left(1 + \sum_{i=1}^{\infty} u_i \lambda^i\right) = 1 + \lambda \sum_{i=1}^{\infty} u_i \lambda^{i-1} \equiv 1 \mod \lambda, \]

which, together with the lemma, implies \( u(\lambda)x_i(\lambda) \equiv 1 \mod \lambda^{2^i} \) (how?).
Let \( \iota(n) \) denote the number of algebraic operations required to compute the first \( n \) coefficients of the inverse of a power series.

**Theorem** If \( 4M(n) \geq M(2n) \geq 2M(n) \) (which we assume for today’s lecture), then \( \iota(n) = O(M(n)) \).

**Proof** Since

\[
\mathbf{u}(\lambda) = \sum_{i=0}^{\infty} u_i \lambda^i = u_0 \left( 1 + \sum_{i=1}^{\infty} \frac{u_i}{u_0} \lambda^i \right),
\]

\[
\mathbf{u}^{-1}(\lambda) = u_0^{-1} \left( 1 + \sum_{i=1}^{\infty} \frac{u_i}{u_0} \lambda^i \right)^{-1}.
\]

(Why may we assume that \( u_0 \neq 0 \)?)

Therefore, the first \( n \) coefficients of \( \mathbf{u}^{-1}(\lambda) \) can be computed from the first \( n \) coefficients of \( \left( 1 + \sum_{i=1}^{\infty} \frac{u_i}{u_0} \lambda^i \right)^{-1} \) in \( n \) additional divisions (by \( u_0 \)), and we may assume that \( u_0 = 1 \).
Let
\[ u^{-1}(\lambda) = \sum_{i=0}^{\infty} x_i \lambda^i. \]

That is, we have to compute (the coefficients of) \( \sum_{i=0}^{n} x_i \lambda^i \).

It follows from the lemma that for each positive integer \( m \),
\[ \sum_{i=0}^{2m} x_i \lambda^i \equiv 2 \sum_{i=0}^{m} x_i \lambda^i - \left( \sum_{i=0}^{2m} u_i \lambda^i \right) \left( \sum_{i=0}^{m} x_i \lambda^i \right)^2 \mod \lambda^{2m}. \]

Therefore,
\[ \iota(n) \leq \iota(n/2) + 5M(n) \]
(why?), implying
\[\begin{align*}
\nu(n) & \leq \nu(n/2) + 5M(n) \\
& \leq \nu(n/4) + 5M(n) + 5M(n/2) \\
& \leq 5 \sum_{i=0}^{\log_2 n} M(n/2^i) \\
& \leq 5 \sum_{i=0}^{\log_2 n} M(n)/2^i \\
& \leq 10M(n).
\end{align*}\]
Computing quotient and remainder for polynomials
Let \( u(\lambda) \) and \( v(\lambda) \) be an \( n \)- and an \( m \)-degree polynomials, respectively, \( m \leq n \):

\[
  u(\lambda) = \sum_{i=0}^{n} u_i \lambda^i \quad \text{and} \quad v(\lambda) = \sum_{i=0}^{m} v_i \lambda^i .
\]

We shall compute the polynomials \( q(\lambda) \) and \( r(\lambda) \) of degree \( n - m \) and less than \( m \), respectively, such that

\[
  u(\lambda) = q(\lambda)v(\lambda) + r(\lambda) .
\]

That is, \( q(\lambda) \) and \( r(\lambda) \) are the quotient and the remainder, respectively, of division of \( u(\lambda) \) by \( v(\lambda) \).

\[
  q(\lambda) = \sum_{i=0}^{n-m} q_i \lambda^i \quad \text{and} \quad r(\lambda) = \sum_{i=0}^{\deg r(\lambda)} r_i \lambda^i .
\]
Then
\[ \hat{u}(\lambda) = q(\lambda) \hat{v}(\lambda) + r(\lambda)\lambda^{n - \deg r(\lambda)} \]
(why?),

implying
\[ \hat{u}(\lambda) \left( \hat{v}(\lambda) \right)^{-1} = q(\lambda) + r(\lambda)\lambda^{n - \deg r(\lambda)} \left( \hat{v}(\lambda) \right)^{-1}. \]
\[ \hat{u}(\lambda) \left( \hat{v}(\lambda) \right)^{-1} = \hat{q}(\lambda) + \hat{r}(\lambda) \lambda^{n - \deg r(\lambda)} \left( \hat{v}(\lambda) \right)^{-1} \]

Since \( \deg \hat{q}(\lambda) = n - m \) and \( \deg \hat{r}(\lambda) < m \), the coefficients of \( \hat{r}(\lambda) \lambda^{n - \deg r(\lambda)} \left( \hat{v}(\lambda) \right)^{-1} \) appear after the coefficients of \( \hat{q}(\lambda) \) in the above sum.

Thus, for computing \( q(\lambda) \) it suffices to compute the first \( n - m \) coefficients of \( \hat{u}(\lambda) \left( \hat{v}(\lambda) \right)^{-1} \).

For this we compute in \( O(M(n)) \) operations the first \( n - m \) coefficients of \( \left( \hat{v}(\lambda) \right)^{-1} \) and then multiply the corresponding polynomial by \( \hat{u}(\lambda) \), which can be done in additional \( O(M(n)) \) operations.

Another \( O(M(n)) \) operations are needed for computing

\[ \hat{r}(\lambda) \lambda^{n - \deg r(\lambda)} = \hat{u}(\lambda) - \hat{q}(\lambda) \hat{v}(\lambda) . \]

Thus, the whole computation (of what?) can be done in \( O(M(n)) \) operations.
Polynomial evaluation
Let $x(\lambda) = \sum_{i=0}^{n-1} x_i \lambda^i$ and let $t_0, t_1, \ldots, t_{n-1}$ be pairwise distinct elements in the domain.

We shall compute $\{x(t_0), x(t_1), \ldots, x(t_{n-1})\}$ (which can be thought of as the components of $W(t_0, t_1, \ldots, t_{n-1})(x_0, x_1, \ldots, x_{n-1})^T$).

We denote the number of operations required for this computation by $E(n)$ and as always we assume that $n$ is a power of 2.

As we have already seen (where?)

$$x(t) = \text{rem}(x(\lambda), \lambda - t).$$

That is, we have to compute

$$x(t_i) = \text{rem}(x(\lambda), \lambda - t_i), \quad i = 0, 1, \ldots, n - 1.$$
For $0 \leq k < \ell < n$, we define the polynomials $y_{k,\ell}(\lambda)$ by

$$y_{k,\ell}(\lambda) = \prod_{i=k}^{\ell-1} (\lambda - t_i).$$

Then

$$y_{k,\ell}(\lambda)y_{\ell,m}(\lambda) = y_{k,m}(\lambda),$$

cf. computing symmetric functions in Lecture 3 (what was the algorithm?). Like in that lecture, all $y_{(j-1)2^\ell,j2^\ell}, \ell = 1, 2, \ldots, \log_2 n - 1$, $j = 1, 2, \ldots, 2^{\log_2 n - \ell}$, can be computed in $O(M(n) \log n)$ operations.

Let

$$r_{k,\ell}(\lambda) = \text{rem}(x(\lambda), y_{k,\ell}(\lambda)).$$

Then

$$x(t_i) = r_{i,i+1}(\lambda), \quad i = 0, 1, \ldots, n - 1,$$

and, for $k' \leq k < \ell < \ell'$,

$$r_{k,\ell}(\lambda) = \text{rem}(r_{k',\ell'}(\lambda), y_{k,\ell}(\lambda))$$

(why?).
It follows that, given $y_{k,\ell}(\lambda)$’s,

$$E(n) = 2E(n/2) + O(M(n))$$

(how?), implying

$$E(n) = O(M(n) \log n)$$

(how?).
Polynomial interpolation
Let \((t_0, u_0), (t_1, u_1), \ldots, (t_{n-1}, u_{n-1})\) be pairs of the domain elements such that \(t_0, t_1, \ldots, t_{n-1}\) are pairwise distinct.

We shall compute the coefficients \(x_0, x_1, \ldots, x_{n-1}\) of the polynomial

\[
x(\lambda) = \sum_{i=0}^{n-1} x_i \lambda^i
\]

such that \(x(t_i) = u_i, i = 0, 1, \ldots, n - 1\) (which can be thought of as the components of \(W^{-1}(t_0, t_1, \ldots, t_{n-1})(u_0, u_1, \ldots, u_{n-1})^T\)).

We denote the number of operations required for this computation by \(\nu(n)\) and as always we assume that \(n\) is a power of 2.
Lemma Let $y(\lambda) = \prod_{i=0}^{n-1} (\lambda - t_i)$ and let $y_i(\lambda) = \frac{y(\lambda)}{\lambda - t_i}$, $i = 0, 1, \ldots, n-1$. Then

$$x(\lambda) = \sum_{i=0}^{n-1} \frac{u_i}{y_i(t_i)} y_i(\lambda).$$

Proof Substituting $t_j$, $j = 0, 1, \ldots, n-1$, for $\lambda$ in $\sum_{i=0}^{n-1} \frac{u_i}{y_i(t_i)} y_i(\lambda)$ we obtain

$$\sum_{i=0}^{n-1} \frac{u_i}{y_i(t_i)} y_i(t_j) = \frac{u_j}{y_j(t_j)} y_j(t_j) = u_j$$

(why do we have the first equality?). That is, the values of the degree-$(n-1)$ polynomials $x(\lambda)$ and $\sum_{i=0}^{n-1} \frac{u_i}{y_i(t_i)} y_i(\lambda)$ are the same at $n$ pairwise distinct points. Therefore, the polynomials are equal (why?). $\square$
\[ y'(\lambda) = \sum_{i=0}^{n-1} \frac{y(\lambda)}{\lambda - t_i} = \sum_{i=0}^{n-1} y_i(\lambda). \]

Therefore,

\[ y_i(t_i) = y'(t_i), \]

\[ i = 0, 1, \ldots, n - 1 \ (\text{why?}). \]

**Bookkeeping**

- The coefficients of \( y(\lambda) = \prod_{i=0}^{n-1} (\lambda - t_i) \) can be computed in \( O(M(n) \log n) \) operations (how?).

- The coefficients of \( y'(\lambda) \) can be computed from (the coefficients of \( y(\lambda) \) in \( O(n) \) operations (how?).

- All values \( y'(t_i), i = 0, 1, \ldots, n-1 \), can be computed in \( O(M(n) \log n) \) operations (how?).
• All values $v_i = \frac{u_i}{y_i(t_i)}$, $i = 0, 1, \ldots, n-1$, can be computed in $n$ operations (how?).

• Finally,

$$x(\lambda) = \sum_{i=0}^{n-1} \frac{u_i}{y_i(t_i)} y_i(\lambda) = \sum_{i=0}^{n-1} v_i y_i(\lambda) = \sum_{i=0}^{n-1} v_i \frac{y(\lambda)}{\lambda - t_i} = y(\lambda) \sum_{i=0}^{n-1} \frac{v_i}{\lambda - t_i}.$$  

That is, $x(\lambda)$ is the numerator of $\sum_{i=0}^{n-1} \frac{v_i}{\lambda - t_i}$ (why?) that can be computed as follows.

For $0 \leq k < \ell < n$, let $n_{k,\ell}(\lambda)$ be the numerator of $\sum_{i=k}^{\ell-1} \frac{v_i}{\lambda - t_i}$. Then $x(\lambda) = n_{0,n}(\lambda)$ and, for $k < \ell < m$,

$$n_{k,m}(\lambda) = y_{\ell,m}(\lambda)n_{k,\ell}(\lambda) + y_{k,\ell}(\lambda)n_{\ell,m}(\lambda).$$
\[ n_{k,m}(\lambda) = y_{\ell,m}(\lambda)n_{k,\ell}(\lambda) + y_{k,\ell}(\lambda)n_{\ell,m}(\lambda) \]

\[ x(\lambda) = n_{0,n}(\lambda) \]

Let \( \sigma(n) \) denote the number of operations need for computing \( n_{0,n}(\lambda) \). Then

\[ \sigma(n) = 2\sigma(n/2) + O(M(n)) \]

(why?), implying \( \sigma(n) = O(M(n) \log n) \) (how?).

Summing up the number of operations at all stages of computation, we obtain \( \iota(n) = O(M(n) \log n) \) (how?). In particular, over \( \mathbb{R} \) and \( \mathbb{C} \), \( \iota(n) = O(n \log^2 n) \) (why?).