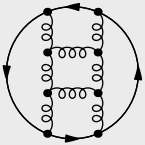
Five-Loop Calculations with FORM

Andreas Maier

6 May 2025

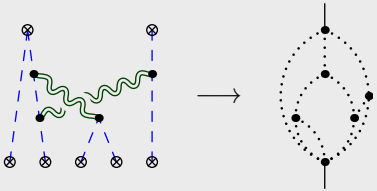
Five-Loop calculations

Massive vacuum diagrams



- QCD decoupling
- Moments of vacuum polarisation
- Anomalous dimensions

Massless propagators



- Gravitational potential

Setup

TOols for MUltiloop CAlculations

1. Generate diagrams: `QGRAF` (Fortran) [Nogueira 1991]

Setup

TOols for MUltiloop CAlculations

1. Generate diagrams: `QGRAF` (Fortran) [Nogueira 1991]
2. Identify diagram families: `autopsy` or `dynast` (Rust),
   based on `nauty` and `Traces` (C) [McKay, Piperno 2014]

Setup

TOols for MUltiloop CAlculations

1. Generate diagrams: `QGRAF` (Fortran) [Nogueira 1991]
2. Identify diagram families: `autopsy` or `dynast` (Rust), based on `nauty` and `Traces` (C) [McKay, Piperno 2014]
3. Manipulate expressions: `nucalc` (FORM)
   - Calculate colour factors: `color` (FORM) [van Ritbergen, Schellekens, Vermaseren 1998]
   - Insert Feynman rules (FORM)
   - Compute traces (FORM)
   - Expand in momenta/masses: `series` (FORM)
   - Reduce tensor integrals: `nucalc` (FORM) or `tide` (Rust)
   - Expand in ϵ: `series` (FORM)
   - Cancel scalar products: `nucalc` (FORM)

Setup

TOols for MUltiloop CAlculations

1. Generate diagrams: `QGRAF` (Fortran) [Nogueira 1991]
2. Identify diagram families: `autopsy` or `dynast` (Rust), based on `nauty` and `Traces` (C) [McKay, Piperno 2014]
3. Manipulate expressions: `nucalc` (FORM)
   - Calculate colour factors: `color` (FORM) [van Ritbergen, Schellekens, Vermaseren 1998]
   - Insert Feynman rules (FORM)
   - Compute traces (FORM)
   - Expand in momenta/masses: `series` (FORM)
   - Reduce tensor integrals: `nucalc` (FORM) or `tide` (Rust)
   - Expand in ϵ: `series` (FORM)
   - Cancel scalar products: `nucalc` (FORM)
4. Reduce to basis integrals: `crusher` + `tinbox` (C++)

Setup

TOols for MUltiloop CAlculations

1. Generate diagrams: `QGRAF` (Fortran) [Nogueira 1991]
2. Identify diagram families: `autopsy` or `dynast` (Rust), based on `nauty` and `Traces` (C) [McKay, Piperno 2014]
3. Manipulate expressions: `nucalc` (FORM)
   - Calculate colour factors: `color` (FORM) [van Ritbergen, Schellekens, Vermaseren 1998]
   - Insert Feynman rules (FORM)
   - Compute traces (FORM)
   - Expand in momenta/masses: `series` (FORM)
   - Reduce tensor integrals: `nucalc` (FORM) or `tide` (Rust)
   - Expand in ϵ: `series` (FORM)
   - Cancel scalar products: `nucalc` (FORM)
4. Reduce to basis integrals: `crusher` + `tinbox` (C++)
5. Compute master integrals

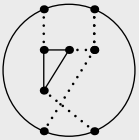
Setup

TOols for MUltiloop CAlculations

1. Generate diagrams: `QGRAF` (Fortran) [Nogueira 1991]
2. Identify diagram families: `autopsy` or `dynast` (Rust), based on `nauty` and `Traces` (C) [McKay, Piperno 2014]
3. Manipulate expressions: `nucalc` (FORM)
   - Calculate colour factors: `color` (FORM) [van Ritbergen, Schellekens, Vermaseren 1998]
   - Insert Feynman rules (FORM)
   - Compute traces (FORM)
   - Expand in momenta/masses: `series` (FORM)
   - Reduce tensor integrals: `nucalc` (FORM) or `tide` (Rust)
   - Expand in ϵ: `series` (FORM)
   - Cancel scalar products: `nucalc` (FORM)
4. Reduce to basis integrals: `crusher` + `tinbox` (C++)
5. Compute master integrals
6. Renormalise (FORM)

Which parts are hard?

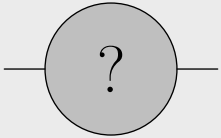
Decoupling / gluon propagator



Step	# Output Terms [10^6]	CPU Time [1000 s]
Insert Feynman Rules	25	1
Compute Traces	5	21
Expand in q	23	4
Reduce Tensor Integrals	7	0.5
Expand in ϵ	3	3
Cancel Scalar Products	2	1

Which parts are hard?

Gravitational potential beyond static sources

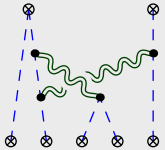


Step	# Output Terms [10^6]	CPU Time [1000 s]
Insert Feynman Rules	76	???
Expand in ϵ	6	???
Expand in velocity	2	???
Resolve energy integrals	1	< 583
Cancel Scalar Products	2	$\lesssim 120$

Hardest family so far does not finish within a few months

Gravitational potential

Energy integrals



- $\propto \int dt_i \frac{d^d p}{(2\pi)^d} e^{i\vec{p}\vec{x}_a(t_i) - ip_0 t_i} \left[1 + \mathcal{O}\left(v_a(t_i)^2\right)\right]$
- Evaluate energy integrals:

$$\int \frac{dp_0}{(2\pi)^d} p_0^n e^{ip_0(t_i - t_j)} \longrightarrow \delta^{(n)}(t_i - t_j)$$

- Repeated integration by parts in $t_i \longrightarrow$ single integral

$$\int dt \int \frac{d^{d-1}q}{(2\pi)^{d-1}} e^{i\vec{q}(x_1(t) - x_2(t))} V\left(\vec{q}, x_a(t), v_a(t), a_a(t), \dots\right)$$

Time-consuming integration by parts over dummy variables

My Wishlist

- Traces
- Index relabelling via graph canonisation [cf. Symbolica]
- Parallelisation diagnostics
- Quality of life, especially encapsulation

Traces

Built-in rules

1. Traces with odd number of γ vanish
2. Replace $\gamma_\mu \gamma^\mu \to \delta^\mu_\mu$, $\not{p}^2 \to p^2$
3. Find pairs $\gamma_\mu \cdots \gamma^\mu$ or $\not{p} \cdots \not{p}$ with minimum distance.
   - Repeatedly anticommute until next to each other
   - Apply rule 2.

Can generate large intermediate expressions:
$25 \cdot 10^6$ terms $\to 361 \cdot 10^6$ terms $\to 5 \cdot 10^6$ terms

Traces

Potential Improvements

- More fine-grained control, e.g. setting $p \cdot q \to 0$ after anticommuting

Traces

Potential Improvements

- More fine-grained control, e.g. setting $p \cdot q \to 0$ after anticommuting
- Sort early, sort often!

Traces

Potential Improvements

- More fine-grained control, e.g. setting $p \cdot q \to 0$ after anticommuting
- Sort early, sort often!
- Rotate to canonical form: $\mathrm{tr}(\gamma_{\mu_2} \cdots \gamma_{\mu_n} \gamma_{\mu_1}) \to \mathrm{tr}(\gamma_{\mu_1} \cdots \gamma_{\mu_n})$
  $\hookrightarrow$ facilitate cancellations between different terms

Traces

Potential Improvements

- More fine-grained control, e.g. setting $p \cdot q \to 0$ after anticommuting
- Sort early, sort often!
- Rotate to canonical form: $\mathrm{tr}(\gamma_{\mu_2} \cdots \gamma_{\mu_n} \gamma_{\mu_1}) \to \mathrm{tr}(\gamma_{\mu_1} \cdots \gamma_{\mu_n})$
  $\hookrightarrow$ facilitate cancellations between different terms
- Work on multiple traces in same term concurrently
- Order of traces (“label”) in same term is irrelevant

Traces

Potential Improvements

- More fine-grained control, e.g. setting $p \cdot q \to 0$ after anticommuting
- Sort early, sort often!
- Rotate to canonical form: $\mathtt{tr}(\gamma_{\mu_2} \cdots \gamma_{\mu_n} \gamma_{\mu_1}) \to \mathtt{tr}(\gamma_{\mu_1} \cdots \gamma_{\mu_n})$
  $\hookrightarrow$ facilitate cancellations between different terms
- Work on multiple traces in same term concurrently
- Order of traces ("label") in same term is irrelevant
- Work towards uniform length of traces in different terms: start with longest traces
- Interleave index renumbering: $\gamma_\mu \gamma_\rho \gamma^\mu - \gamma_\nu \gamma_\rho \gamma^\nu = 0$

Index Renumbering

Lots of dummy indices / variables:

- Feynman rules for vector and tensor boson vertices
- Time integrals in gravitational potential
- Loop momenta

Full canonisation in FORM:

If there are N sets of dummy indices all N! permutations are tried. This can be very costly [...]

Index Renumbering

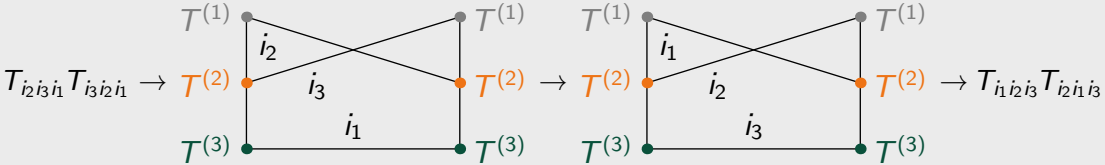
Lots of dummy indices / variables:

- Feynman rules for vector and tensor boson vertices
- Time integrals in gravitational potential
- Loop momenta

Full canonisation in FORM:

If there are N sets of dummy indices all N! permutations are tried. This can be very costly [...]

Alternative: graph canonisation [cf. Symbolica]



- Straightforward parallelisation one of the most important (T)FORM features
- `moduleoption` annotations needed for dollar variables:

  FORM will veto the use of multiple threads/processors for modules in which dollar variables obtain values [...]

  - Helpful diagnostics: which dollar variables block parallel execution?
  - `on parallel diagnostics;`?

Parallelisation

Better encapsulation

Need extra code to execute procedure with dollar variables in parallel

```
l foo =
#include- large_expression
;
bracket 'F',ep;
.sort
keep brackets;
#call expand('F',ep,'CUT')
* ...maybe more code here ...
#call parallel
.sort
```

- Breaks encapsulation: users should not have to care about internal dollar variables
- Nonlocality: `moduleoption` code far away from actual dollar variables, harder to understand and get right

Parallelisation

Better encapsulation

Need extra code to execute procedure with dollar variables in parallel

```
l foo =
#include- large_expression
;
bracket 'F',ep;
.sort
keep brackets;
#call expand('F',ep,'CUT')
* ...maybe more code here ...
#call parallel
.sort
```

- Breaks encapsulation: users should not have to care about internal dollar variables
- Nonlocality: `moduleoption` code far away from actual dollar variables, harder to understand and get right

Ideas:

- Allow `moduleoption local $...;` and similar earlier in module?
- Alternative: make `local` etc. part of dollar variable declaration?
  $\hookrightarrow$ next slides

Quality of Life

Encapsulation

Example from `nucalc` code:

```
#do i=1,‘$num‘P’’
   #$reduction‘i’ = [nucalcReduceScalarProducts::sp](‘P’‘i’,‘P’‘i’) - [nucalcReduceScalarProducts::sp](‘$‘P’‘i’’,‘$‘P’‘i’’);
   #inside $reduction‘i’
      id [nucalcReduceScalarProducts::sp]([nucalcReduceScalarProducts::p]?,[nucalcReduceScalarProducts::q]?) = [nucalcReduceScalarProd
      #do L1={‘?L’}
         #do L2={‘?L’}
            id ‘L1’.‘L2’ = [nucalcReduceScalarProducts::sp](‘L1’, ‘L2’);
         #enddo
      #enddo
   #endinside
#enddo

#define EQS "$reduction1"
#do i=2,‘$num‘P’’
   #redefine EQS "‘EQS’,$reduction‘i’"
#enddo

#call RSPSolveLinear([nucalcReduceScalarProducts::sp],‘EQS’)
```

- Evil hack: Perl script creating package-local variable name from `[:sp]` etc.
- Manual prefix for package-internal procedure: `RSPSolveLinear`
- Both original and generated code hard to read

Quality of Life

Encapsulation

```
#package PACKAGE

* code here

#endpackage
```

and then

- name mangling

```
#package PACKAGE

  symbols x,y,z;

  #procedure PROC
  #endprocedure

  #define $var (local)

#endpackage

#call PACKAGE@PROC
#importprocedure PROC = PACKAGE@PROC
#call PROC
```

Quality of Life

Encapsulation

```
#package PACKAGE

* code here

#endpackage
```

and then

- name mangling
- or new keyword for internal objects, cf. `static` in C/C++
  ↪ hard to implement for variables

Quality of Life

Asking for the Moon

Conclusions

- FORM is a central ingredient in five-loop setup:
  - Colour factors
  - Feynman rules, index contractions
  - Traces
  - Series expansions
  - ...
- Looking forward to future developments