# Creative Barkeeping

## or: How can AD be explained without jargon?

## and: Season Greetings by AD

Uwe Naumann

Software and Tools for Computational Engineering,[1]
RWTH Aachen University, Aachen, Germany

---

[1] https://www.stce.rwth-aachen.de/

# Problem

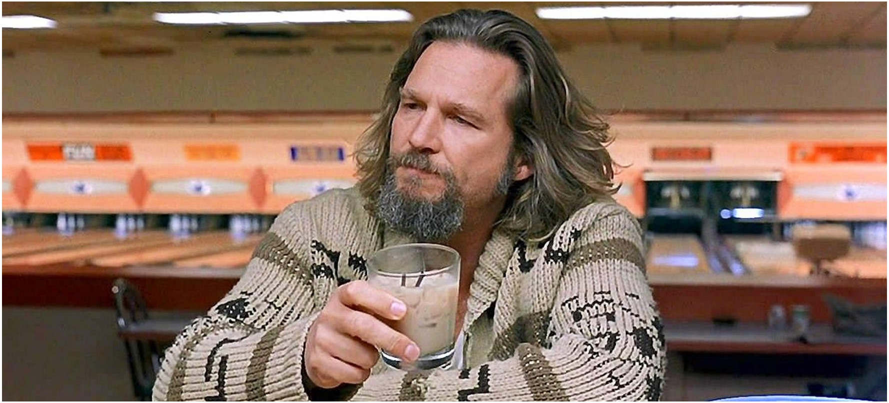
Have been looking for a common story for teaching

- Introduction to C++ (1st semester BSc)
- Introduction to AD (3rd/5th semester BSc)
- Advanced AD (5th semester BSc / MSc)

for a while.

Aiming for shortest path to playing chain rule on directed acyclic graphs.

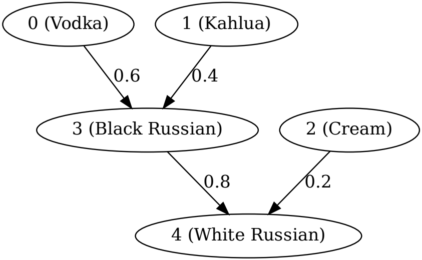
# Solution



“Careful Man. There’s A Beverage Here!”

# Recipes

```
void recipe_cost(
  const std::vector<mixable> &x,
  std::vector<mixable> &y
) {
  y[0]=0.6*x[0]+0.4*x[1];
  y[1]=0.8*y[0]+0.2*x[2];
}
```



$$x = \begin{pmatrix} \text{Vodka} \\ \text{Kahlua} \\ \text{Cream} \end{pmatrix}$$

$$y = \begin{pmatrix} \text{Black Russian} \\ \text{White Russian} \end{pmatrix}$$

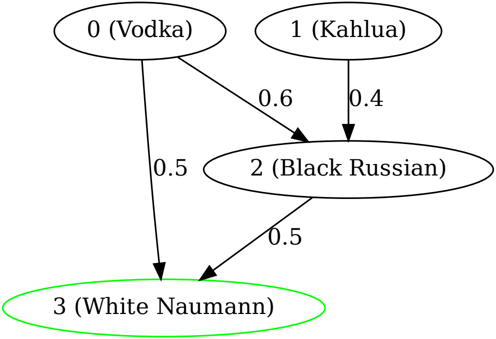
for given mixing ratios, e.g., Black Russian contains 60% Vodka and 40% Kahlua.

$$x = \begin{pmatrix} 30 \\ 20 \\ 5 \end{pmatrix} \quad \Rightarrow \quad y = \begin{pmatrix} 26 \\ 21.8 \end{pmatrix}$$

# Disclaimer

Alcoholic beverages are not a crucial prerequisite for the conceptual soundness of the upcoming story!

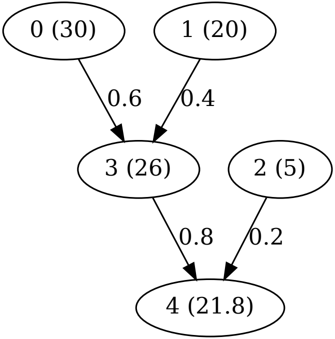
# BTW ...



# Pricing (a Recipe)

E.g., for unit cost of Vodka = 30, Kahlua = 20, and Cream = 5 :

Note: Recipe cost is linear and, hence, self-tangent. (Not a problem ...)

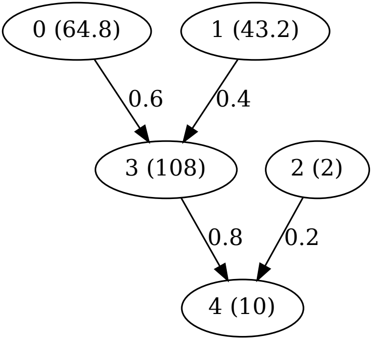


Tangent:

$$\dot{v}_j = \sum_{i \in P_j} \frac{\partial v_j}{\partial v_i} \cdot \dot{v}_i$$

# Stocking (for a Recipe)

E.g., 100 Black and 10 White Russians:



Adjoint:

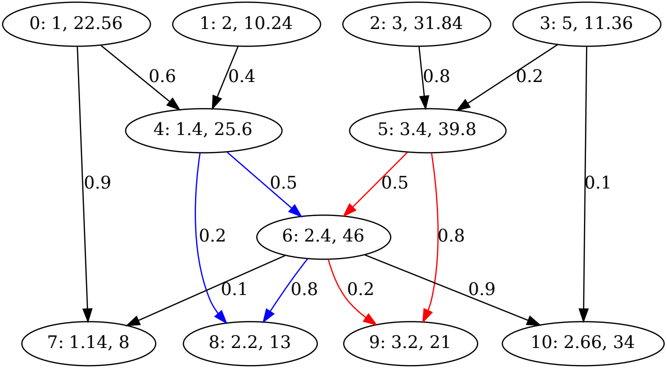
$$\bar{v}_i = \bar{v}_i + \sum_{j \in S_i} \bar{v}_j \cdot \frac{\partial v_j}{\partial v_i}$$

## Consequences, e.g., ...

- Vector modes for recipe cost and recipe stock
  ... vector tangent and adjoint AD [1]
- Compression of sparse recipe summaries
  ... Jacobian compression [2]
- NP-hardness of RECIPE SUMMARY
  ... JACOBIAN ACCUMULATION [3]
- NP-hardness of RECIPE STOCK
  ... DAG REVERSAL [4]
- Checkpointing recipe stock code
  ... checkpointing adjoint code [5]
- (Generalized) Elimination techniques for RECIPE SUMMARY
  ... AD MISSION PLANNING [6]

# And Yes, ...

... solutions for RECIPE SUMMARY may require *face elimination* [7]!

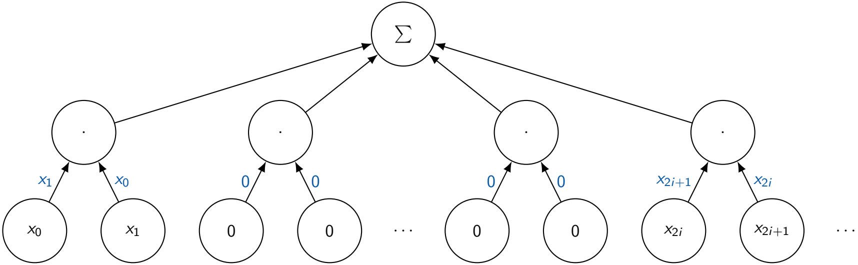


# Outlook

Next week ...

# AD wishes Merry Xmas

$$y = \sum_{i=0}^{n-1} x_{2i} \cdot x_{2i+1}$$



Live ...

# AD wishes Merry Xmas

AD is used to implement type-generic tangent and adjoint versions of

$$y = \sum_{i=0}^{n-1} x_{2i} \cdot x_{2i+1}$$

in C++. Instantiations with the data type of **x** equal to **char** and output of the gradient at $(101\ 77\ 114\ 114\ 32\ 121\ 109\ 88\ 115\ 97)^T$ to `std::cout` yields

Merry Xmas.

Zeros can be added to the input vector to explore the significantly varying run times of the different derivative codes while not modfiying its output.

# AD wishes Happy 2026

$$y = \frac{1}{6} \cdot \sum_{i=0}^{n-1} x_i^3$$

$$\frac{d^2 y}{dx^2} = \begin{pmatrix} x_0 & 0 & \cdots & 0 \\ 0 & x_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{n-1} \end{pmatrix}$$

Live ...

# AD wishes Happy 2026

Type-generic sparsity-aware second-order tangent and second-order adjoint versions of

$$y = \frac{1}{6} \cdot \sum_{i=0}^{n-1} x_i^3$$

yield

Happy 2026

at $(72\ 97\ 112\ 112\ 121\ 32\ 50\ 48\ 50\ 54)^T$.

Zeros can be added to the input vector to explore the significantly varying run times of the different derivative codes while not modfiying its output.

# Conclusion

Refer to

U.N.: Seasons's Greetings by AD. arXiv:2402.09409v2 [cs.CE], 2025.

and

`https://github.com/un110076/SeasonsGreetings`.

for the full story.

Merry Xmas and Happy 2026!

# References

[1] A. Griewank and A. Walther, Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, no. 105 in Other Titles in Applied Mathematics, SIAM, 2nd ed., 2008.

[2] A. Gebremedhin, F. Manne, and A. Pothen, What color is your Jacobian? Graph coloring for computing derivatives, SIAM Review, 47 (2005), pp. 629–705.

[3] U. N., Optimal Jacobian accumulation is NP-complete, Math. Program., 112 (2008), pp. 427–441.

[4] U. N., DAG reversal is NP-complete, J. Discr. Alg., 7 (2009), pp. 402–410.

[5] A. Griewank, Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation, Opt. Meth. Softw., 1 (1992), pp. 35–54.

[6] U. N., E. Schneidereit, S. Märtens, and M. Towara, Elimination techniques for algorithmic differentiation revisited, in SIAM Conference on Applied and Computational Discrete Algorithms (ACDA23), 2023, pp. 201–212.

[7] U. N., Optimal accumulation of Jacobian matrices by elimination methods on the dual computational graph, Math. Program., 99 (2004), pp. 399–421.