ZACHARY DEL ROSARIO

MANAGING UNCERTAINTY: A PRIMER

MANAGING UNCERTAINTY – A PRIMER

TUFTE-LATEX.GITHUB.IO/TUFTE-LATEX/

First compiling, January 2018

Contents

Introduction

William Gibson's masterful short story *The Gernsback Continuum*[1] follows a man haunted by an idea. His protagonist is pursued by *semiotic ghosts*, lingering packets of meaning, there in the form of ray-gun spacemen and ten-engine airships sprung from the mind of legendary science fiction author Hugo Gernsback. This character's dilemma is Gibson's reaction to the starry-eyed view of science fiction in the 1970's – a view stuck in the chrome-plated, streamlined future of the 1930's which could never be.

We as engineers in the early 2000's are haunted by a different semiotic ghost, that of an engineering certainty which never existed. Our tools are predicated on a deterministic, omniscent view of the universe which is woefully at odds with the reality of our occupation. Our traditional approach is to make uncertainty irrelevant – to pile on margin and arbitrary safety factors until we creep away from failure, at the expense of inefficient, expensive designs, and a lack of understanding of the relevant uncertainties.

This book is an attempt at exoricism. Rather than making it irrelevant, we aim to *Manage Uncertainty*.

Dummy citation.[2]

[1] William Gibson. *Burning Chrome*. Orion Publishing Group, 1986

[2] S. Shvartsman. personal communication

Sensitivity Analysis

Motivation

Here at Stanford,[3] we host the Predictive Science Academic Alliance Program II (PSAAP II). This project is motivated by a solar receiver intended for green energy production, with heat transfer enhanced via a particle-laden working fluid. This is a system of enormous complexity, featuring particle-laden turbulent flow with radiative heat transfer. The difficulty arises (in part) due to the complex multiphysics, which necessitates careful modeling for computational tractability. For example, the Stanford PSAAP II simulation codes implement models not just for the turbulence, but also novel models for radiation transport, accounting for distributed particles smaller than the resolved gridsize.

[3] Circa 2018

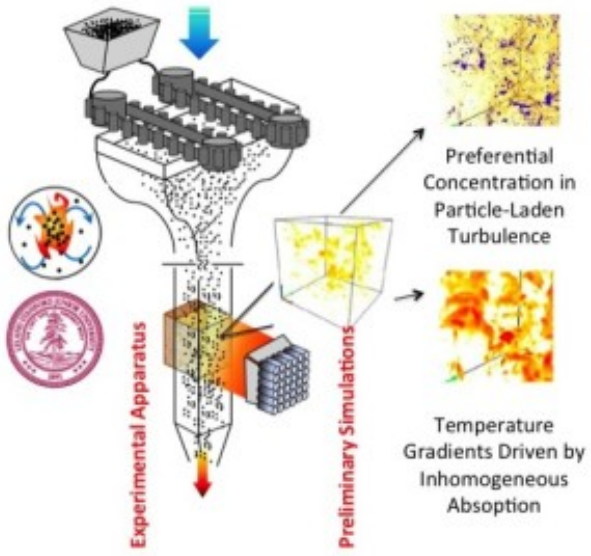


Figure 1: Schematic for the PSAAP II physical problem. This graphic highlights some of the complex interactions between traditionally separate domain physics. The complexity of the physics necessitates modeling assumptions, which introduce uncertainties.

Note that some of the techniques we will discuss are sample-intensive. Most of the techniques described below are intended for *computer experiments*; that is, simulation campaigns run on computer codes. Contrast these with *physical experiments*, in which the physical world 'solves' the equations of motion automatically.

These modeling assumptions introduce non-physical coefficients, which constitute additional uncertainties in the problem. Table 1 lists these and other examples of uncertainties in the PSAAP II problem. For practical purposes, we would be interested in how *all* the uncertainties affect practical output quantities of interest, such as the device efficiency. If the results were largely insensitive to our uncertainties over a range of physically relevant values, this would build confidence in our ability to perform accurate simulations, and would potentially enable the design of novel solar receivers.

Type	Source
Input Uncertainties	Turbulence model parameters
Model Discrepancies	Limitations of eddy viscosity
Numerical Errors	Artificial dissipation
Experimental Uncertainties	Probe accuracy limitations

Table 1: A non-exhaustive list of examples of uncertainties in the Stanford PSAAP II problem.

If our outputs were *not* insensitive to these uncertainties, that would signal that we need to either reduce the uncertainties (say through experimentation), or account for the uncertainties in our results (say through uncertainty propagation). This would necessitate a combination of experimental and computational work – both are expensive activities.

This expense is compunded by the *curse of dimensionality*, which states (roughly) that the expense of a procedure[4] tends to grow *exponentially* with dimension. Not only is gathering data expensive; we need many data points to understand high-dimensional parameter spaces!

[4] e.g. a parameter study, or numerical quadrature

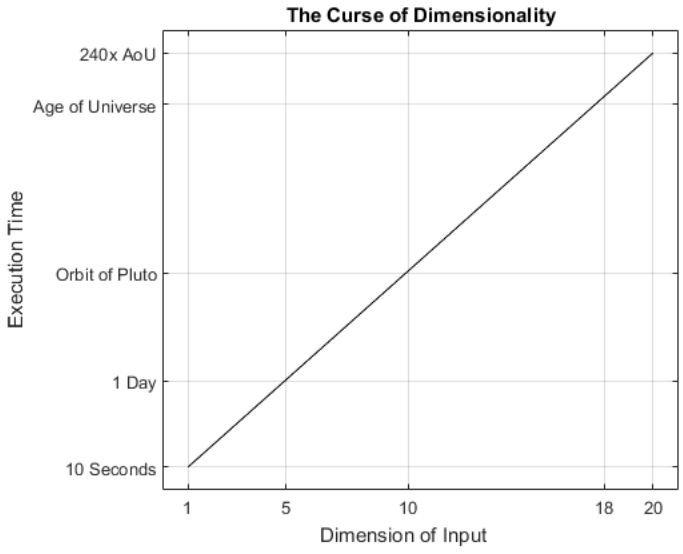


Figure 2: Cartoon depicting computational expense under the curse of dimensionality. Imagine that we are performing quadrature (numerical integration), and are using a simple tensor-product approach; i.e. using 10 evaluations per dimension. If a computer implementation of our qoi evaluates in one second, it will take 10 seconds to study a 1-dimensional function. For a 5-dimensional function, this will take a day to evaluate. This exponential growth quickly grows out of hand, reaching an execution time of nearly the Age of the Universe at just 18 dimensions.

Ideally, one would like to know which parameters are most in-

fluential, preferably before embarking on an expensive simulation campaign. This is the insight that sensitivity analysis seeks to provide. A sensible way to address this curse is to attack dimensionality directly – to reduce the number of input parameters. As a simple approach, we could perform a sensitivity analysis, identify those parameters which do not appreciably affect our qoi, and freeze them to nominal values. This effectively reduces the dimensionality of the problem, ameliorating the curse of dimensionality.

An intermediate conclusion

Sensitivity analysis is about determining how input parameters affect a chosen quantity of interest. We would apply these results at multiple stages in an analysis. A sensitivity analysis could help identify which uncertainties are worth characterizing through a physical experiment. Such an analysis could also identify parameters which can be frozen to nominal values, making further computational experiments less expensive.

Local vs Global

There are two broad philosophies to sensitivity analysis: local and global. Local sensitivity generally refers to studying the gradient of a qoi; the derivative (partial or total) of the output qoi with respect to its input parameters, computed at a set of nominal values. Global sensitivity generally refers to variance contributions; the amount of variance exhibited in the output qoi, as contributed by different sets of the input parameters.

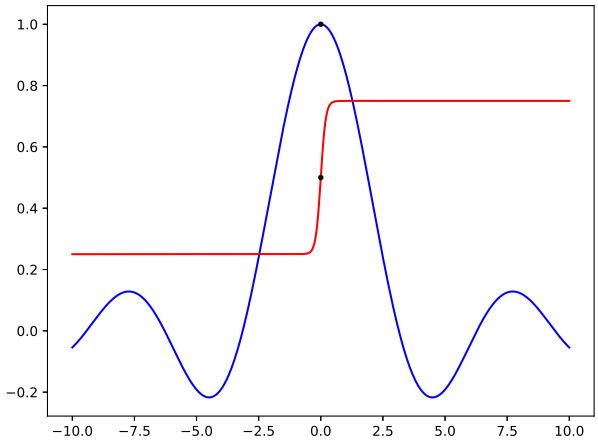


Figure 3: Cartoon functions to depict differences between local and global sensitivity. The vertical axis corresponds to our qoi, while the horizontal corresponds to an input parameter. The black dot is the point of interest for local sensitivity. The blue curve has a great deal of global sensitivity, but zero local sensitivity. The red curve has comparatively little global sensitivity, but a large local sensitivity. Which metric is important depends on the objective of the study.

As implied by the names, the two approaches give different pieces of information. Depending on the context, one or the other may

be the 'right' metric to study. Figure 3 provides a simple cartoon illustration of two cases where local and global sensitivities will be in disagreement.

In what follows, we will focus on local sensitivity, and leave global sensitivity to a following set of notes.

Preliminary Considerations

Before we jump into computing the derivative, we will first discuss some preliminary concerns. Before computing the derivative, we ought to determine *where* we ought to evaluate! Furthermore, we ought to have some clear goal in mind of what to *do* with the gradient before we set out to compute it.

To introduce some notation, let $f(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$ be our quantity of interest (qoi), and let $\Omega \subseteq \mathbb{R}^d$ be the parameter space of interest. The first object we'll study is Ω.

Region identification

The parameter space Ω may itself be uncertain, or at least poorly characterized. It is possible that insights gained in one subset of Ω may not hold in other regions, or modeling choices which are appropriate in one region may fail in another. Figure 4 depicts a classical example from fluid mechanics where physical behavior changes dramatically in different regions of parameter space. The punchline here is that one should think carefully about the domain Ω to be studied, as this will affect the conclusions drawn from sensitivity analysis, either local or global.

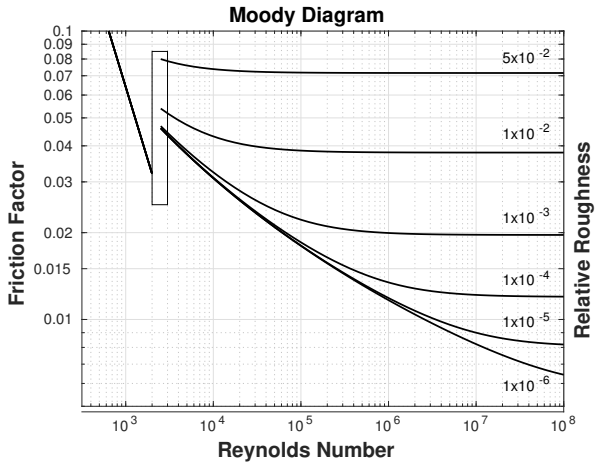


Figure 4: The Moody Diagram depicts the dimensionless pressure losses (Friction Factor) against relevant dimensionless parameters. Note that the Relative Roughness only becomes relevant at high Reynolds Number. This sort of information would not be discovered through a global sensitivity analysis, and instead requires careful probing of local properties.

Pseudoglobal approach

Studying the gradient at every point in parameter space is both expensive and conceptually challenging in high dimensions. Visually inspecting a high dimensional space is not possible, and generally requires some form of dimension reduction for visualization.

One could generate a summary of local behavior by computing an expectation of the gradient, for example

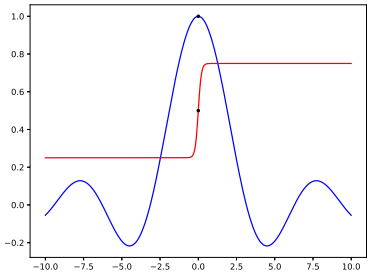
$$\mathbb{E}\left[\left|\frac{\partial f}{\partial x_i}\right|\right], \tag{1}$$

which can be approximated via Monte Carlo sampling.[5] This quantity could be considered a global average of local effects, mixing the two philosophies. The textbook (Ch. 6) describes this as a *pseudoglobal* approach. **??** is closely related to *Morris screening*, a global sensitivity approach we will discuss in the next set of notes, and is also discussed in Chapter 15 of the textbook.[6]

If (1) were small for some set of indices $I \subseteq \{1, \ldots, d\}$, we could freeze the associated variables to nominal values $\boldsymbol{x}_I = \boldsymbol{x}_{I,\text{Nom}}$. This would reduce the dimensionality of the problem. Note that a purely local approach would *not* endorse this pick-and-freeze approach, as illustrated by the blue curve in Figure 3, repeated in the margin. Furthermore, this pseudoglobal approach is prone to miss nonlinear behavior in the function – imagine if we had sampled outside the sharp region of the red curve in Figure 3. A pseudoglobal approach is not necessarily the best way to attack the problem of freezing variables; this is better handled via global sensitivity analysis.

[5] Monte Carlo is a technique for approximating integrals, which can be viewed as expectations against some integral weight, possibly uniform. One proceeds by drawing samples $\boldsymbol{x}_i \sim \rho$ according to the integral weight, and estimating $\mathbb{E}[f(\boldsymbol{x})] \approx \frac{1}{n}\sum f(\boldsymbol{x}_i)$.

[6] Ralph C. Smith. *Uncertainty quantification: theory, implementation, and applications*. SIAM, 2013



Determining the stability of an optimal point

Unconstrained optimization is conceptually straightforward; critical points are found where the gradient of the qoi is zero. Optimal points are then found where the curvature meets an additional criterion.[7]The magnitude of the curvature gives some sense of how sensitive the optimal point is to changes in parameter values.

For constrained optimization, the intuition is slightly different. Rather than having zero gradient, a function must meet a stationarity condition; its gradient must 'balance' that of the constraints.[8] Figure 5 gives a cartoon example of a constrained optimization problem. The punchline is that the gradient of our qoi need not be zero for a constrained optimal point. This is not a big deal if our parameters are known exactly.

However, if our parameters are uncertain, a nonzero gradient provides some information about how much this uncertainty can affect our optimal value. As a more concrete example in an engineering

[7] e.g. for a maximum, we want negative curvature

[8] More formally, the Karush-Kuhn-Tucker (KKT) conditions give a set of first-order necessary conditions for optimality.

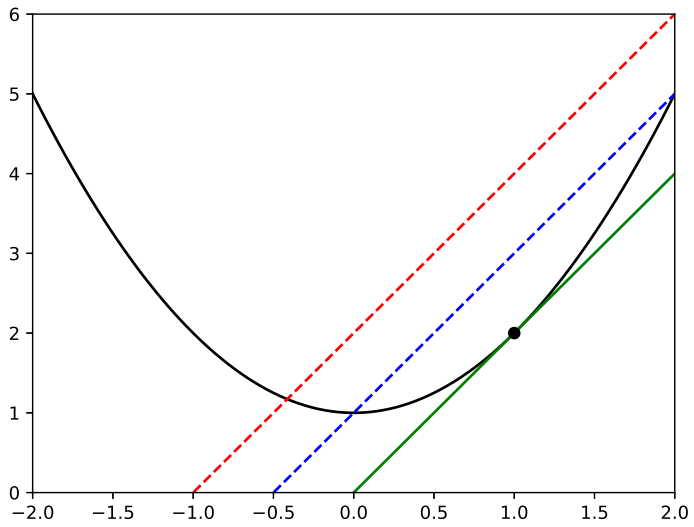


Figure 5: Cartoon of constrained optimization: The black curve is the constraint, while the red, blue, and green lines are isocontours of the qoi. If the gradient of our qoi is not aligned with the gradient of the constraint at their intersection – if they are not *tangent* – then one can find an improved function value by sliding along the constraint. Here, the green isocontour is tangent at the black dot, which corresponds to an optimal point. The gradient of the qoi is nonzero at this point; if there is uncertainty in the input parameters, this could lead to the realized value of the response being significantly less than desired. One may quantify these effects through local sensitivity analysis

design context, one could do a Taylor approximation to determine how much the qoi may fluctuate, then add margin to the design to account for this uncertainty.

Finite Differencing

Conceptually, approximating the derivative is quite simple. One simply chooses a base point $\boldsymbol{x}$ and considers a finite step-size Δx approximation to the derivative[9]

$$\frac{df}{dx_i} \approx \frac{f(x_1, \ldots, x_i + \Delta x, \ldots, x_d) - f(\boldsymbol{x})}{\Delta x}. \tag{2}$$

However, there are some complications with this approach. First, this computational procedure requires additional evaluations, with a cost that grows linearly with dimensionality d. If we require N evaluations of the gradient for a desired procedure (e.g. building a map of the gradient or performing quadrature), then our total cost will be $O(Nd)$.

The other complication is choosing a sensible Δx. Recent work has studied the selection of Δx based on measuring the *empirical noise* of a function; that is, tailoring the step-size based on the observered variability in a computed qoi arising from a computational procedure.[10]

[9] Note that this is the *partial derivative*, which is usually denoted with ∂. We're going to reserve the ∂ symbol for a different operation in these notes.

[10] Jorge J Moré and Stefan M Wild. Estimating derivatives of noisy simulations. *ACM Transactions on Mathematical Software (TOMS)*, 38(3):19, 2012

Adjoint Method

The adjoint is clever method for cheaply computing the sensitivity of a scalar qoi. This is particularly useful if evaluating our qoi requires solving some parameterized governing equation $F(\boldsymbol{y}, \boldsymbol{x})$ for a state variable (e.g. flow field) on which our qoi depends $f(\boldsymbol{y}, \boldsymbol{x})$. Practically, it allows us to compute the total derivative of our qoi $\frac{df}{d\boldsymbol{x}}$ at the expense of one additional solution comparable to $F(\boldsymbol{y}, \boldsymbol{x})$. Steven Johnson [11] has some very nice introductory notes on the topic, which I follow for the next example. He also points to a paper by Cao et al.[12] which gives a more general treatment of the adjoint.

[11] Steven G. Johnson. Notes on adjoint methods for 18.335, 2012

[12] Yang Cao, Shengtai Li, Linda Petzold, and Radu Serban. Adjoint sensitivity analysis for differential-algebraic equations: The adjoint dae system and its numerical solution. *SIAM Journal on Scientific Computing*, 24(3):1076–1089, 2003

Example: Linear System

The spirit of the adjoint method is well-illustrated by a simple linear system. Suppose we have a governing equation for the state $A_{jk}(\boldsymbol{x})y_k = b_j(\boldsymbol{x})$ [13], where our equation is parameterized by $\boldsymbol{x}$. Our qoi is some known function of the state and our parameters $f(\boldsymbol{y}, \boldsymbol{x})$. By chain rule, the sensitivity is

[13] I'm going to use Einstein notation here to help make this easier to follow.

$$\frac{df}{dx_i} = \frac{\partial f}{\partial x_i} + \frac{\partial f}{\partial y_j}\frac{\partial y_j}{\partial x_i}. \tag{3}$$

Since f is known, the partials $\frac{\partial f}{\partial x_i}$ and $\frac{\partial f}{\partial y_j}$ are easy to evaluate.[14] However, the quantity $\frac{\partial y_j}{\partial x_i}$ is more challenging. Using the matrix inverse, we have $y_j = A^{-1}_{jk}b_k$. Taking the partial, we have[15]

[14] Here, we use ∂ to denote a 'direct' derivative. For example, if $f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=j}^{m} y_j$, then $\frac{\partial f}{\partial x_i} = 0$ and $\frac{\partial f}{\partial y_j} = 1$.

[15] Here, we use the identity $\frac{\partial \boldsymbol{A}^{-1}}{\partial s} = -\boldsymbol{A}^{-1}\frac{\partial \boldsymbol{A}}{\partial s}\boldsymbol{A}^{-1}$. One can derive this by re-arranging $\frac{\partial(\boldsymbol{A}^{-1}\boldsymbol{A})}{\partial s}$.

$$\begin{aligned}
\frac{\partial y_j}{\partial x_i} &= \frac{\partial A^{-1}_{jk}}{\partial x_i}b_k + A^{-1}_{jk}\frac{\partial b_k}{\partial x_i}, \\
&= A^{-1}_{jk}\left(-\frac{\partial A_{kl}}{\partial x_i}A^{-1}_{lp}b_p + \frac{\partial b_k}{\partial x_i}\right), \\
&= A^{-1}_{jk}\left(-\frac{\partial A_{kl}}{\partial x_i}x_l + \frac{\partial b_k}{\partial x_i}\right).
\end{aligned} \tag{4}$$

Substituting into (3) yields

Using Einstein notation makes it clear in (5) that the quantity $\frac{\partial A_{kl}}{\partial x_i}$ is a rank 3 tensor, as there are three indices.

$$\frac{df}{dx_i} = \frac{\partial f}{\partial x_i} + \frac{\partial f}{\partial y_j}A^{-1}_{jk}\left(-\frac{\partial A_{kl}}{\partial x_i}y_l + \frac{\partial b_k}{\partial x_i}\right). \tag{5}$$

One could evaluate (5) by carrying out the matrix multiplication (which would be expensive), or by carrying out the product $\frac{\partial f}{\partial y_j}A^{-1}_{jk} \equiv \lambda_k$. This quantity is the solution of the *adjoint equation*

$$\lambda_k A_{kj} = \frac{\partial f}{\partial y_j}. \tag{6}$$

In summary, to compute the sensitivity via the adjoint method, we need (1) the solution to the governing equation $\boldsymbol{y}$ at our chosen parameter values, (2) the partial derivatives to known functions

$\frac{\partial f}{\partial x_i}$, $\frac{\partial f}{\partial y_j}$, $\frac{\partial A_{kl}}{\partial x_i}$, $\frac{\partial b_k}{\partial x_i}$, and (3) the solution to the adjoint equation λ_k. We then evaluate the total derivative via

$$\frac{df}{dx_i} = \frac{\partial f}{\partial x_i} + \lambda_k A^{-1}_{jk} \left(-\frac{\partial A_{kl}}{\partial x_i} y_l + \frac{\partial b_k}{\partial x_i} \right). \tag{7}$$

Conclusion

This set of notes introduced the motivation behind sensitivity analysis, the broad philosophies of local and global approaches, some uses of sensitivity analysis, and approaches for performing local sensitivity analysis. Next time, we'll dig into techniques for performing global sensitivity analysis.

Appendix

Some useful derivations

Personally, I found some of the claims in Johnson's notes rather mysterious.[16] I attempt to demystify some of those points in this section.

[16] Steven G. Johnson. Notes on adjoint methods for 18.335, 2012

The derivative of a matrix exponential ends up being a convolution. This is because the derivative is more easily carried out in the frequency domain, and we pick up a convolution from doing so. Let $\boldsymbol{A} = \exp(-t\boldsymbol{B})$. First, we'll work out the Laplace transform of this quantity.

$$\mathcal{L}\{\boldsymbol{A}\}(s) = (s\boldsymbol{I} + \boldsymbol{B})^{-1}. \tag{8}$$

Next, we'll take the derivative, using the identify $\frac{\partial \boldsymbol{A}^{-1}}{\partial x} = -\boldsymbol{A}^{-1} \frac{\partial \boldsymbol{A}}{\partial x} \boldsymbol{A}^{-1}$.

$$\frac{\partial}{\partial x}\mathcal{L}\{\boldsymbol{A}\}(s) = -(s\boldsymbol{I} + \boldsymbol{B})^{-1} \frac{\partial \boldsymbol{B}}{\partial x} (s\boldsymbol{I} + \boldsymbol{B})^{-1}. \tag{9}$$

Finally, we'll take an inverse Laplace transform,[17] which yields

[17] Assuming we can commute the derivative with the integral

$$\begin{aligned} \frac{\partial \boldsymbol{A}}{\partial x} &= -\left[e^{-\boldsymbol{B}t} \frac{\partial \boldsymbol{B}}{\partial x} \right] \circledast \left[e^{-\boldsymbol{B}t} \right], \\ &= -\int_0^t e^{-\boldsymbol{B}t'} \frac{\partial \boldsymbol{B}}{\partial x} e^{-\boldsymbol{B}(t-t')} dt' \end{aligned} \tag{10}$$

This enters into the derivative equation via

$$\frac{df}{dx_i}(t) = \frac{\partial f}{\partial x_i}(t) + \lambda_k(t) \left(-\frac{\partial A_{kl}}{\partial x_i}(t) y_l(t) + \frac{\partial y_k}{\partial x_i}(0) \right). \tag{11}$$

In Johnson's notes, he states a form equivalent to

$$\frac{df}{dx_i}(t) = \frac{\partial f}{\partial x_i}(t) + \int_0^t \lambda_k(t-t') \frac{\partial \boldsymbol{B}}{\partial x_i} y_l(t') dt' + \lambda_k(t) \frac{\partial y_k}{\partial x_i}(0). \tag{12}$$

This varies from what one might expect as the 'plug in' result from manipulating (11). However, recall that the matrix exponential acts as a shift operator – it *shifts* initial data forward (or backward) in time to give the solution at a desired time. Commuting terms and applying the shifts results in (12) above.

Bibliography

[1] Yang Cao, Shengtai Li, Linda Petzold, and Radu Serban. Adjoint sensitivity analysis for differential-algebraic equations: The adjoint dae system and its numerical solution. *SIAM Journal on Scientific Computing*, 24(3):1076–1089, 2003.

[2] William Gibson. *Burning Chrome*. Orion Publishing Group, 1986.

[3] Steven G. Johnson. Notes on adjoint methods for 18.335, 2012.

[4] Jorge J Moré and Stefan M Wild. Estimating derivatives of noisy simulations. *ACM Transactions on Mathematical Software (TOMS)*, 38(3):19, 2012.

[5] S. Shvartsman. personal communication.

[6] Ralph C. Smith. *Uncertainty quantification: theory, implementation, and applications*. SIAM, 2013.