LLNL-TR-807158

LAWRENCE LIVERMORE NATIONAL LABORATORY

Free-field Ground Motion Induced by Underground Explosions at Aqueduct Mesa with Predictions for Physical Experiment One (PE1)

S. R. Ford

March 13, 2020

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

Free-field Ground Motion Induced by Underground Explosions at Aqueduct Mesa with Predictions for Physical Experiment One (PE1)

Sean R. Ford

Lawrence Livermore National Laboratory, Livermore, California 94550, sean@llnl.gov

March 13, 2020

Abstract

We fit standard power-law models for ground acceleration and particle velocity using scaled ground motion observations from underground nuclear explosions at Aqueduct Mesa on the Nevada National Security Site. The models are then validated using small-scale high-explosive (HE) test data from nearby Rainier Mesa. The comparison between model predictions and observations is good and can be made more favorable if the assumed yield for the HE tests are doubled. Further validation is made with a numerical experiment that additionally shows a transition in ground motion attenuation to elastic propagation ($1/r$). An extended model is provided to incorporate the long-range transition in attenuation, which results in increased ground motions at farther range. We then make predictions for the field experiment, Physical Experiment One (PE1), to take place in P-tunnel on Aqueduct Mesa, and suggest that the models could be used in planning various operations for PE1.

1 Introduction

The U.S. conducted underground nuclear tests at Aqueduct Mesa on the Nevada National Security Site (formerly, the Nevada Test Site) [Department of Energy, 2015]. More information on the tests conducted there can be found in Townsend et al. [2007].

Fourney et al. [1994] and Ingram and Drake [1987] provide scaled ground motion observations of the events at Aqueduct Mesa. The events considered are listed in Table 1 and the data is shown in Figure 1. We use the reported values of velocity and scaled acceleration

Table 1: Aqueduct Mesa events in the dataset

Name	Tunnel
PLATTE	K
MISSION CYBER	P
DISKO ELM	P
DISTANT ZENITH	P
MINT LEAF	T
DIAMOND SCULLS	T
HUSKY PUP	T
MIDAS MYTH / MILAGRO	T

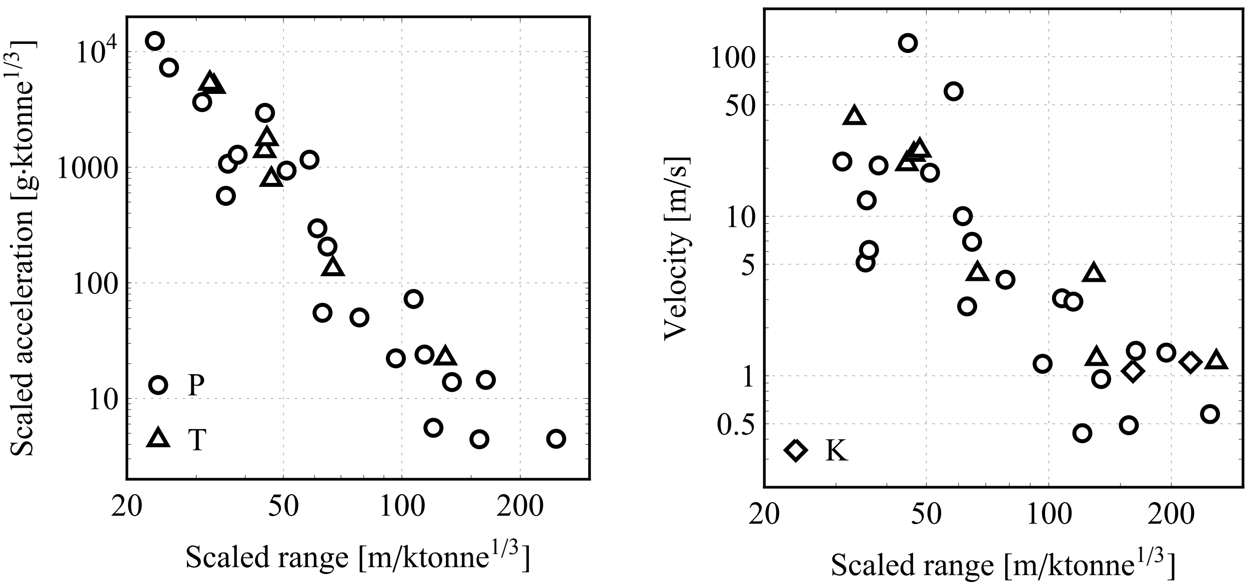

Figure 1: Dataset of a) scaled acceleration and b) velocity observations.

versus scaled range to parameterize scaled ground motion power-law models. These models can be used for ground motion prediction for a given scaled range.

There were also several (chemical) high-explosive (HE) tests at Rainier Mesa and we compare the observations of two such events, One-Ton and Puff-Too [Smith, 1980], with the ground motion predictions derived from underground nuclear tests at Aqueduct Mesa. Aqueduct Mesa is a northern extension of Rainier Mesa and is geologically very similar. The observations compare favorably with the predictions, which gives us confidence that we can make ground motion predictions for the Physical Experiment One (PE1) that is scheduled to take place at Aqueduct Mesa as part of the Low Yield Nuclear Monitoring program.

Table 2: HE tests used for validation

Name	Tunnel
Puff-Too	G
One-Ton	G

2 Inference

We will proceed to fit ground motion models of the form found in Perret and Bass [1975]. Our goal is to estimate parameters α and β in

$$d = \alpha \left(r/w^{1/3}\right)^{\beta}, \tag{1}$$

where r is the slant range and w is the explosive yield in TNT-equivalent ktonne (where tonne is a metric ton equal to 1000 kg), or the energy released in a ktonne of trinitrotoluene (TNT), which is defined as 4.184 $\times 10^{12}$ J, d is the scaled observation, such that scaled acceleration is $a \cdot w^{1/3}$ in g·ktonne$^{1/3}$ and velocity yield scaling cancels and is simply u in m/s. The standard approach used in Perret and Bass [1975] is to transform the power-law relationship in Equation 1 to a log-linear one of the form

$$\log(d) = \log(\alpha) + \beta \log\left(r/w^{1/3}\right), \tag{2}$$

and use simple linear regression techniques that can estimate the parameters a and b in an equation of the form

$$\boldsymbol{y} = a + b\boldsymbol{x} + \boldsymbol{\epsilon}. \tag{3}$$

So for the case presented in Equation 2, the vector of outcomes $\boldsymbol{y}$ is $\log(d_i)$ for each occurrence i and the vector of predictors $\boldsymbol{x}$ is $\log(r_i/w_i^{1/3})$, which is the logarithm of the scaled range in m/ktonne$^{1/3}$. The parameters a and b are $\log(\alpha)$ and β, respectively, and $\boldsymbol{\epsilon}$ is a vector of (normal) random errors.

3 Models

We estimate the parameters using the Mathematica[1] `LinearModelFit` function. Single prediction bands are calculated as $\hat{y} \pm t_c\sqrt{X^{\top} C\, X + \sigma^2}$, where X is the design matrix (made up of a column of ones and the predictor), C is the covariance matrix of the parameters, σ^2 is the variance in the residuals (predicted outcome − observed) estimated from the fit, and

[1] http://www.wolfram.com/mathematica

t_c is the cumulative density of the Students-T distribution for a $c\%$ confidence band.

3.1 Acceleration

The limits of the model can be derived from the data, which in scaled range are 23.5 and 246.8 m/ktonne$^{1/3}$, so lets call the limits 25 to 250 m/ktonne$^{1/3}$, such that outside this range we are *extrapolating*. Within these limits the model is

$$\hat{y}_{\mathrm{acc}} = \log(a \cdot w^{1/3}) = 21.11 - 3.759x \tag{4}$$

where, again, the predictor x is $\log(r/w^{1/3})$. The 95% prediction bands are

$$\hat{y}_{\mathrm{acc}} \pm 2.056\sqrt{1.578 - 0.4667x + 0.05629x^2} \tag{5}$$

The 90 and 99% maximum prediction bands are given by changing the coefficient (derived from the Students-T) from 2.056 to 1.706 and 2.779, respectively. The model and prediction bands are shown in Figure 2a. In the parlance of Perret and Bass [1975], where the variance for $\hat{\beta}$ is reported as a fractional standard deviation of the exponent and the variance for $\hat{\alpha}$ is reported as the anti-log (in other words, $\exp(x)$) of its standard error and its reciprocal, the acceleration model is

$$a \cdot w^{1/3} = 1.4719 \times 10^9 (r/w^{1/3})^{-3.76\pm0.24}, \tag{6}$$

where the coefficient of this equation, which represents the intercept of the fit at $r/w^{1/3}$ equal to unity, has a variance defined by the factors 2.702 and its reciprocal. The exponent has a fractional standard deviation of 6.31%.

3.2 Velocity

The limits of the model are the same as in the acceleration model, which is 25 to 250 m/ktonne$^{1/3}$, such that outside this range we are *extrapolating*. Within these limits the model is

$$\hat{y}_{\mathrm{vel}} = \log(u) = 9.837 - 1.886x \tag{7}$$

where, again, the predictor x is $\log(r/w^{-1/3})$. The 95% prediction bands are

$$\hat{y}_{\mathrm{vel}} \pm 2.045\sqrt{1.964 - 0.5276x + 0.06011x^2} \tag{8}$$

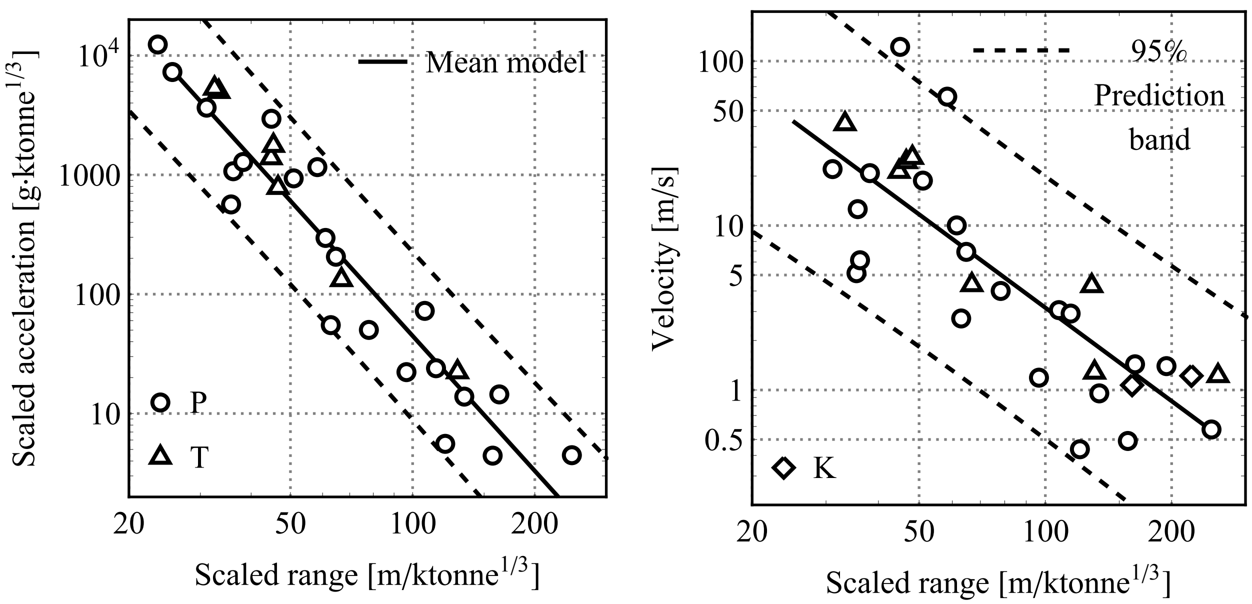

Figure 2: Observations and best-fit model (solid line) with 95% prediction band (dashed line).

The 90 and 99% maximum prediction bands are given by changing the coefficient (derived from the Students-T) from 2.045 to 1.699 and 2.756, respectively. The model and prediction bands are shown in Figure 2b. In the parlance of Perret and Bass [1975], the velocity model is

$$u = 1.8713 \times 10^4 (r/w^{1/3})^{-1.89 \pm 0.25}, \quad (9)$$

where the coefficient has a variance defined by the factors 2.967 and its reciprocal. The exponent has a fractional standard deviation of 12.99%.

4 Discussion

We can compare the Aqueduct Mesa model derived here with the Perret and Bass [1975] wet-tuff model and use both for predictions of the Physical Experiment One (PE1).

4.1 Comparison to Perret and Bass [1975]

Perret and Bass [1975] provide a “Wet Tuff” model with an applicable range of 30 to 600 m/ktonne$^{1/3}$. The acceleration model is

$$a \cdot w^{1/3} = 4.31 \times 10^7 (r/w^{1/3})^{-2.61 \pm 0.17}, \quad (10)$$

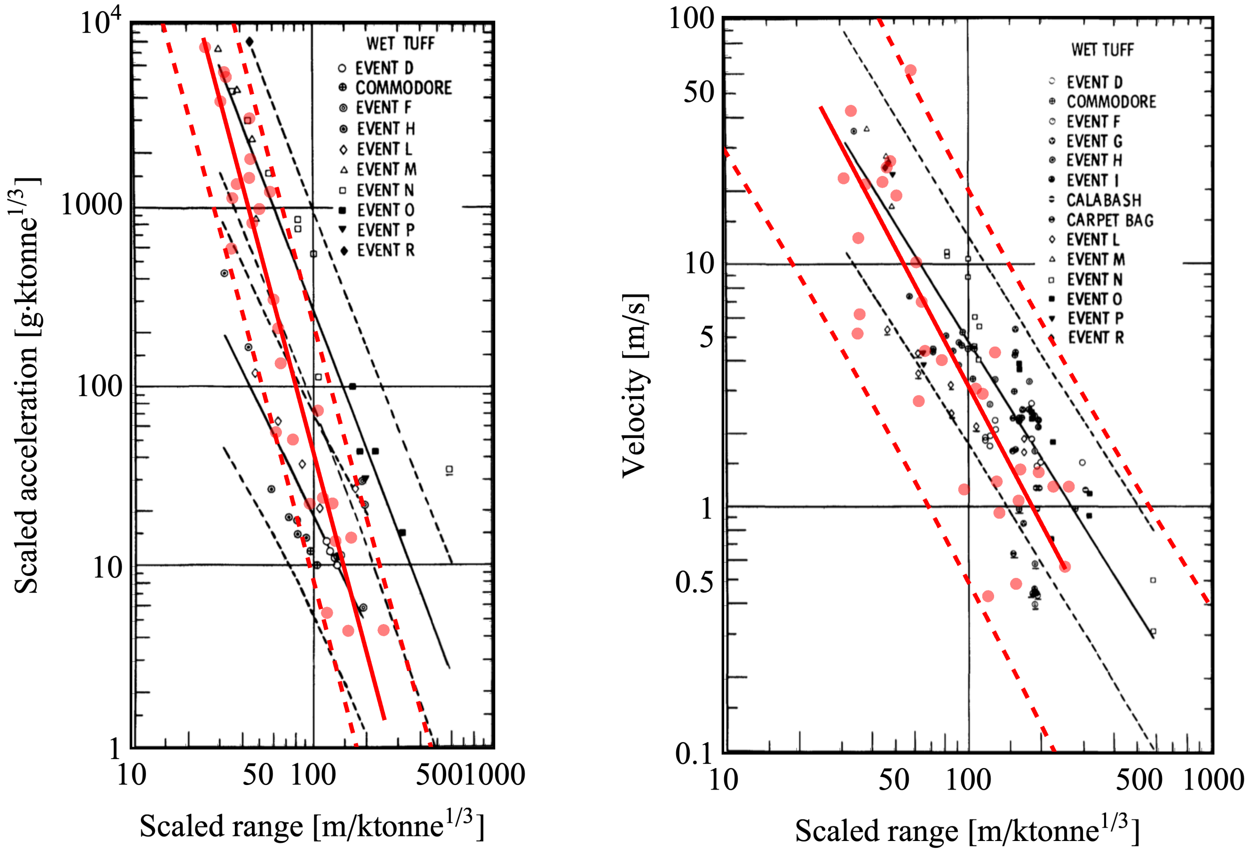

Figure 3: Ground motion models (red line) with 95% prediction bands (dashed red lines) derived from all the Aqueduct Mesa observations (red circles) for acceleration (left) and velocity (right) plotted on top of figures 3.7 (left) and 3.8 (right) from Perret and Bass [1975] showing the "Wet Tuff" ground motion models and observations.

where the coefficient has a variance defined by the factors 2.21 and its reciprocal. The exponent has a fractional standard deviation of 5.8%. The velocity model is

$$u = 6.61 \times 10^3 (r/w^{1/3})^{-1.56\pm0.09}, \tag{11}$$

where the coefficient has a variance defined by the factors 1.56 and its reciprocal. The exponent has a fractional standard deviation of 5.8%. Unfortunately, Perret and Bass [1975] do not provide the prediction variance used to derive error bands nor a function for them, so we cannot compare them quantitatively. However, they are plotted in their report, so we can plot our models and error bands on top of their figures for some measure of qualitative comparison. This is done in Figure 3 for the acceleration and velocity models, which also show the Aqueduct Mesa dataset (red circles).

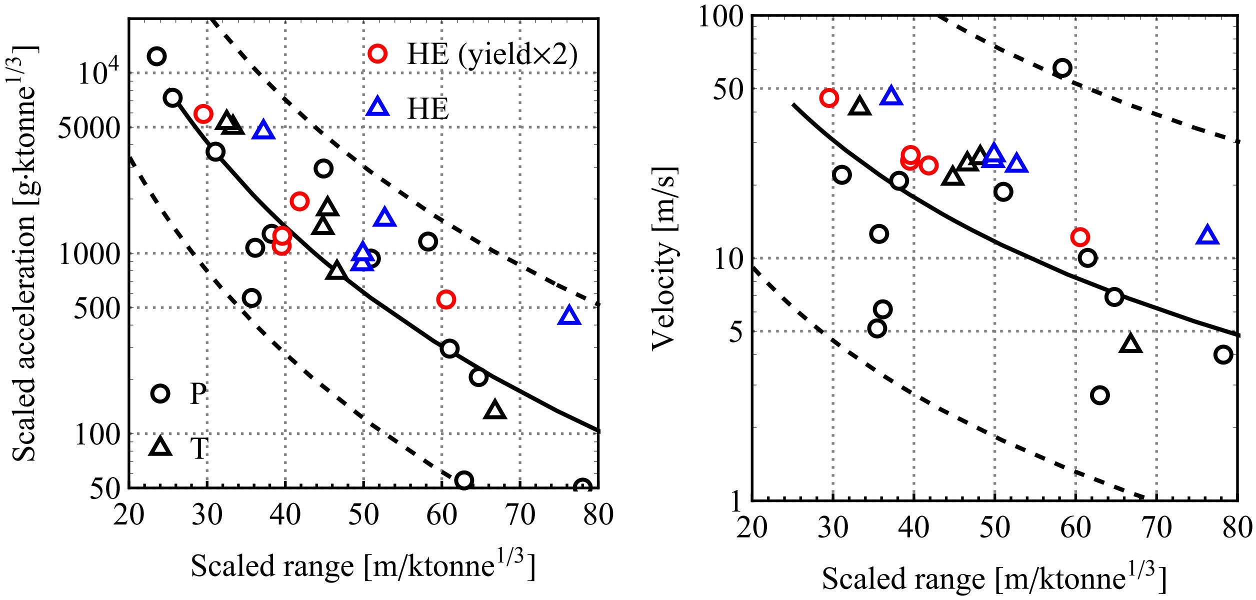

Figure 4: Observations of HE tests (blue symbols) compared with the model derived from the Aqueduct Mesa dataset (black lines with 95% predictive band in dashed lines). There is better agreement if we assume the effective yield is twice the given HE yield (red symbols).

4.2 Validation

We validate the model with datasets that were not used in training from both observations and numerical experiments. With important relevance to PE1, these datasets are from chemical explosives.

4.2.1 Small HE tests

We predict the ground motion for two small HE tests in G-tunnel on Rainier Mesa called Puff-Too and One-Ton. Although, the tests were not on Aqueduct Mesa the high saturation of the rock where these tests took place is similar to some of the Aqueduct Mesa geology so they may provide an appropriate test of the model derived from the Aqueduct Mesa datasets. Figure 4 compares the observations with the model. There is better agreement with the model if the effective yield of the HE datasets is assumed to be two times the given yield (red symbols), which is consistent with observations from the Non-Proliferation Experiment [Goldstein and Jarpe, 1994, Denny, 1994].

4.2.2 Numerical experiments

We calculate the ground motion using Geodyn with a material model for saturated tuff and an explosive model for Comp-B. More details on the experiment can be found in Vitali et al. [2020]. Figure 5 compares some initial numerical results with the model. There is better agreement with the model if the effective yield is assumed to be twice the input yield. We can also see where the regime enters small strains and propagation becomes nearly elastic. This is where the ground motion is inversely proportional to range, which looks to be around 60 m/ktonne$^{1/3}$. Using this insight we amend the ground motion model to change in slope near this point, which is also plotted in Figure 5. The new models take the original dependence as in Equations 4 and 7, but then at a chosen elastic transition point (in this case, scaled range of 150 m/ktonne$^{1/3}$) they change to extended models with $1/r_s$ dependence and the prediction interval is found by extending out the interval at the transition range.

$$\hat{y}_{\text{acc}}^{\text{ext}} = \log(a \cdot w^{1/3}) = \begin{cases} 21.11 - 3.759x & 25 \leq (r/w^{1/3}) < 150 \\ 7.287 - x & 150 \leq (r/w^{1/3}) \leq 400 \end{cases} \tag{12}$$

where, the predictor x is $\log(r/w^{1/3})$. The 95% prediction bands are

$$\hat{y}_{\text{acc}}^{\text{ext}} \pm \begin{cases} 2.056\sqrt{1.578 - 0.4667x + 0.05629x^2} & (r/w^{1/3}) < 150 \\ 1.661 & (r/w^{1/3}) \geq 150 \end{cases} \tag{13}$$

The 90 and 99% maximum prediction bands after the transition are given by changing the interval value from 1.661 to 1.378 and 2.245, respectively. The change for before the transition range are as given after Equation 5. The velocity model is

$$\hat{y}_{\text{vel}}^{\text{ext}} = \log(u) = \begin{cases} 9.837 - 1.886x & 25 \leq (r/w^{1/3}) < 150 \\ 5.398 - x & 150 \leq (r/w^{1/3}) \leq 400 \end{cases} \tag{14}$$

where, again, the predictor x is $\log(r/w^{1/3})$. The 95% prediction bands are

$$\hat{y}_{\text{vel}}^{\text{ext}} \pm \begin{cases} 2.045\sqrt{1.964 - 0.5276x + 0.06011x^2} & (r/w^{1/3}) < 150 \\ 1.863 & (r/w^{1/3}) \geq 150 \end{cases} \tag{15}$$

The 90 and 99% maximum prediction bands after the transition range are given by changing the interval value from 1.863 to 1.547 and 2.510, respectively. The change for before the transition range are as given after Equation 8.

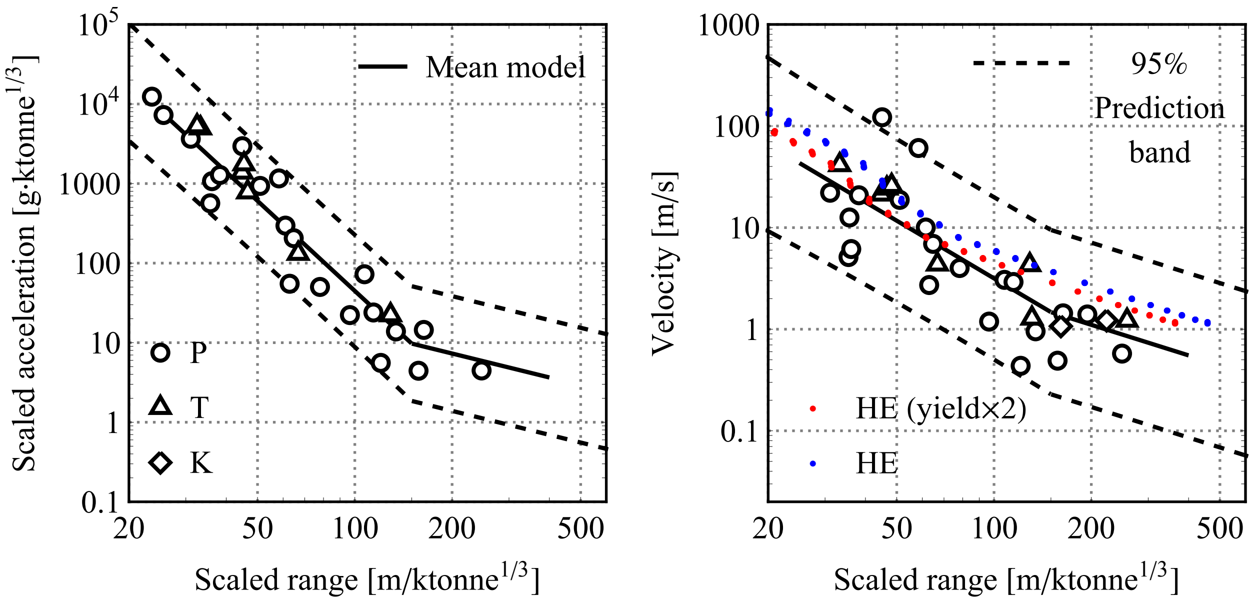

Figure 5: Ground motion prediction from Geodyn calculation. There is better agreement if we assume the effective yield is twice the given HE yield (red symbols). The amended model is also given with a transition to elastic propagation at 150 m/ktonne$^{1/3}$.

4.3 Application to PE1

For a given explosive yield and observation range, Equations 6 and 9 derived from the Aqueduct Mesa dataset allow us to predict ground motion with error. Based on the comparison with HE observations and numerical predictions, we recommend that ground motion for a chemical explosion be predicted using an effective yield that is two times the TNT-equivalent yield.

4.3.1 PE1-A

PE1-A is a fully coupled chemical explosion in P-tunnel. Since the comparison with the small-scale HE tests showed better agreement with the model if the assumed yield is twice the given yield, Figures 6 and 7 give the unscaled values for twice the chemical explosive yield of 18 tonnes TNT-equivalent. The 95% prediction is about 1 g in acceleration and 1 m/s in velocity at about 200 m range. We also consider a transition to elastic propagation at 150 m/ktonne$^{1/3}$. The 95% prediction is now approximately 30 g in acceleration and 3 m/s in velocity at about 200 m range.

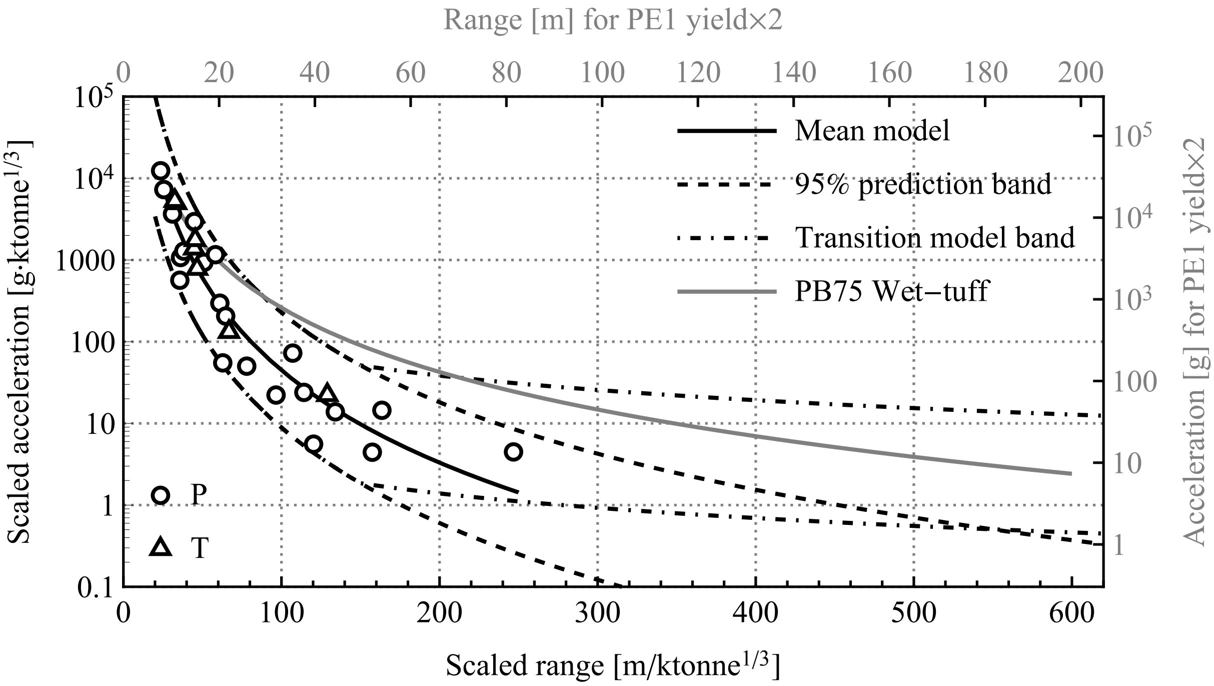

Figure 6: Similar to Figure 2 but linear range axes and with right and top axes as unscaled range and acceleration for two times the chemical explosive yield of PE1-A of 18 tonne TNT-equivalent. The model from Perret and Bass [1975] is shown for comparison (gray line). Additionally, a model that transitions to elastic propagation ($1/r$) at 150 m/ktonne$^{1/3}$ is given (dot-dashed line).

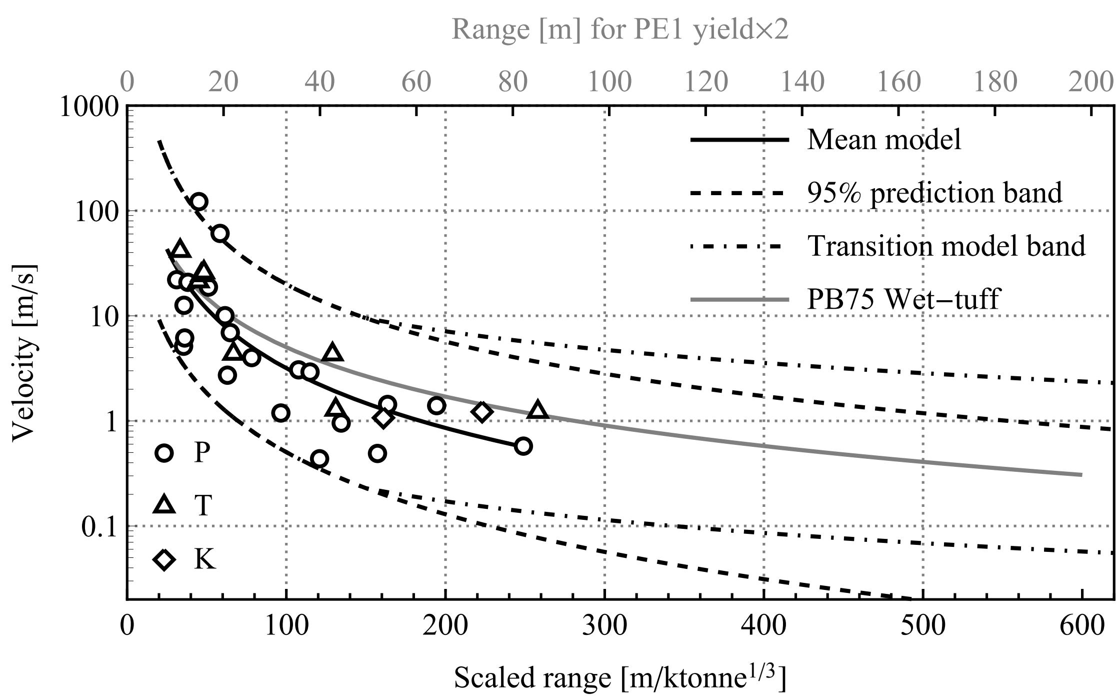

Figure 7: Similar to Figure 2 and Figure 6 but for particle velocity.

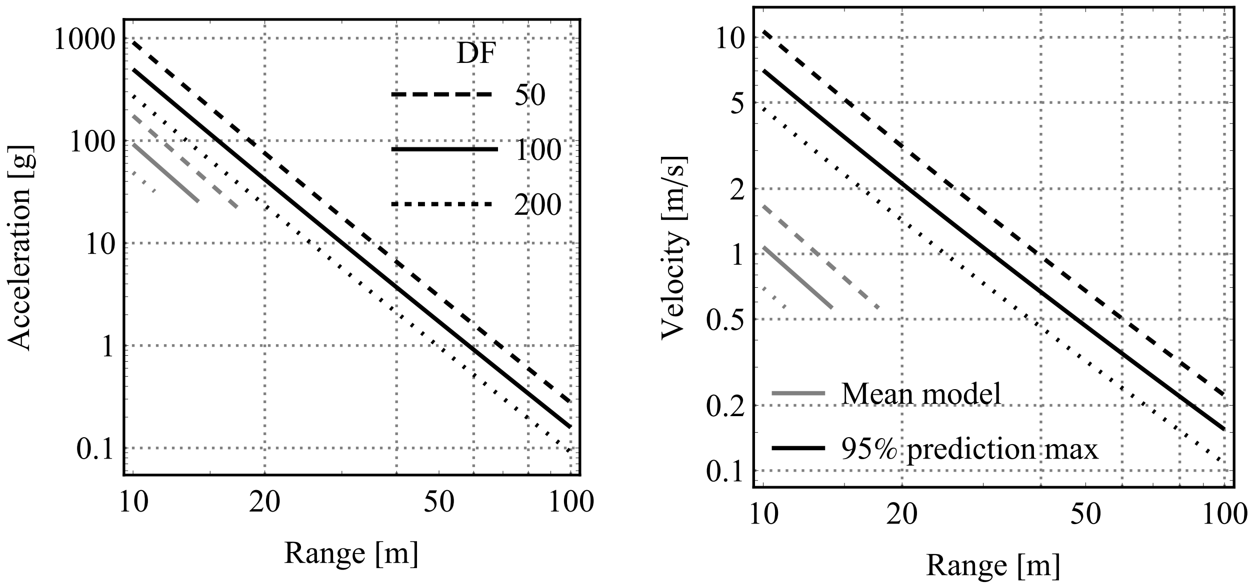

Figure 8: Predicted ground motion for PE1-B with decoupling factors (DF) of 50 (dashed), 100 (solid), and 200 (dotted). The gray lines are the mean model predictions (shown only out to the model range) and the black lines are the maximum 95% prediction interval.

4.3.2 PE1-B

PE1-B is a fully decoupled chemical explosion in a hemispherical cavity in P-tunnel. Figure 8a shows the ground acceleration prediction for both models assuming arbitrary decoupling factors of 50, 100 and 200, which is achieved by decreasing the given yield by that factor. Figure 8b does the same for ground velocity. A conservative approach to ground motion prediction would be to use the maximum and minimum for a given distance.

Acknowledgments

A computer program (in Python3) is provided to calculate ground motion prediction intervals from the Aqueduct Mesa model described herein with the option to allow for a transition to elastic attenuation range dependence ($1/r$).

References

Marvin D. Denny, editor. *Proceedings of the Symposium on the Non-Proliferation Experiment: Results and Implications for Test Ban Treaties*. Department of Energy, Rockville, Maryland, April 1994. https://www.osti.gov/biblio/1095232-proceedings-symposium-non-proliferation-experiment-results-implications-test-ban-treaties-rockville-maryland-april.

Department of Energy. United States Nuclear Tests, July 1945 through September 1992, September 2015. Technical Report DOE/NV–209-REV 16, National Nuclear Security Administration Nevada Field Office, Las Vegas, Nevada, September 2015.

W L Fourney, R D Dick, Steve R Taylor, and T A Weaver. An analysis of three nuclear events in P-Tunnel. Technical Report LA-UR-94-1750, Los Alamos National Laboratory, United States, May 1994.

Peter Goldstein and Stephen P. Jarpe. Comparison of Chemical and Nuclear Explosion Source-Time Functions from Close-in, Local, and Regional Seismic Data. *Arms Control and Nonproliferation Technologies*, 1994. DOE/AN/ACNT-94A.

L. F. Ingram and J. L. Drake. Ground Shock Predictions for Underground Nuclear Explosions in Rainier Mesa (U). Technical Report DNA-TR-88-14, Applied Research Associates, Inc., Albuquerque, New Mexico, October 1987. (Confidential/Formerly Restricted Data).

William R Perret and Robert C Bass. Free-field Ground Motion Induced by Underground Explosions. Technical Report SAND-74-0252, Sandia Laboratories, Albuquerque, New Mexico, February 1975.

C.W. Smith. PUFF TOO: A residual stress experiment. Technical report, Sandia National Laboratories, Albuquerque, New Mexico, April 1980.

Dean R. Townsend, Margaret Townsend, and Byron L. Ristvet. A Geotechnical Perspective on Post-Test Data for Underground Nucler Tests Conducted in Rainier Mesa. Special Report DTRIAC-SR-07-002, Defense Threat Reduction Agency, Kirtland AFB, New Mexico, September 2007. (Distribution Statement C: USG and their contractors only).

Efrem Vitali, Oleg Y. Vorobiev, and Sean R. Ford. Simulations of Decoupled Chemical Explosions in Saturated Tuff. *Journal of Geophysical Research*, 2020. in preparation.

Listing 1: Python3 program to calculate ground motion with given prediction interval

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
#
"""
usage: amgm.py [-h] [-r RANGE] [-w YIELD] [-c CONFIDENCE] [-t TRANSITIONRANGE]

Calculate predicted ground motions for Aquaduct Mesa
using model from Ford (2020) LLNL-TR-807158.

arguments:
  -h, --help            show this help message and exit
  -r RANGE, --range RANGE
                        Range from working point [m] (default: 15.0)
  -w YIELD, --yield YIELD
                        Explosive yield [ktonne TNT-equivalent (4.184e12 J)]
                        (default: 0.01)
  -c CONFIDENCE, --confidence CONFIDENCE
                        Confidence level for prediction [%]
                        Options are 90, 95, and 99 (default: 95)
  -t TRANSITIONRANGE, --transition TRANSITIONRANGE
                        Option to force elastic propagation at the given
                        scaled range [m/ktonne^(1/3)]. Option with no value
                        puts it at default scaled range (default: 150.0)

output: acceleration (g) and velocity (m/s) for specified parameters
        (with % prediction band)

version: 0.5.0

author: Sean R. Ford, sean@llnl.gov

examples:

1) no transition to elastic propagation
% python3 amgm.py --range=15.0 --yield=0.01 --confidence=95
Scaled range = 69.6 m/ktonne^(1/3) in model range (25 to 250)
Ground motion [min to max] (95% single prediction interval)
Acceleration = 162.392 to 4037.31 g (Mean = 809.707 g)
Velocity = 0.996928 to 39.3538 m/s (Mean = 6.26361 m/s)

2) transition to elastic propagation at default scaled range (150 m/ktonne^(1/3))
% python3 amgm.py -r 75.0 -w 0.01 -c95 -t
Scaled range = 348.1 m/ktonne^(1/3) out of model range (25 to 250)
Option chosen to turn on elastic propagation (1/r) at 150.0 m/ktonne^(1/3)
Ground motion [min to max] (95% single prediction interval)
Acceleration = 3.70406 to 102.557 g (Mean = 19.4904 g)
Velocity = 0.0985351 to 4.08789 m/s (Mean = 0.634666 m/s)

3) transition to elastic propagation at user-supplied scaled range
% python3 amgm.py -r 75.0 -w 0.01 -c95 -t 120.0
Scaled range = 348.1 m/ktonne^(1/3) out of model range (25 to 250)
Option chosen to turn on elastic propagation (1/r) at 120.0 m/ktonne^(1/3)
Ground motion [min to max] (95% single prediction interval)
Acceleration = 7.02356 to 185.25 g (Mean = 36.071 g)
Velocity = 0.121942 to 4.90512 m/s (Mean = 0.773393 m/s)
"""


import argparse
from math import exp, log, sqrt


def main():
    parser = argparse.ArgumentParser(
        description=(
            "Calculate ground motions using model derived from " "Aqueduct Mesa dataset"
        ),
        formatter_class=argparse.ArgumentDefaultsHelpFormatter,
    )
    parser.add_argument(
        "-r",
        "--range",
        type=float,
        default=15.0,
```

```
        help="Range from working point [m]",
        dest="rangem",
    )
    parser.add_argument(
        "-w",
        "--yield",
        type=float,
        default=0.01,
        help="Explosive yield [ktonne TNT-equivalent (4.184e12 J)]",
        dest="yieldkton",
    )
    parser.add_argument(
        "-c",
        "--confidence",
        type=int,
        choices=[90, 95, 99],
        default=95,
        help="Confidence level for prediction [%%]",
        dest="confidencepercent",
    )
    parser.add_argument(
        "-t",
        "--transition",
        nargs="?",
        const=150,
        type=float,
        help="Option to turn on elastic propagation (1/r) at the given scaled "
        "range [m/\N{CUBE ROOT}ktonne]. Option with no value turns on elastic "
        "propagation at 150 m/\N{CUBE ROOT}ktonne",
        dest="transitionscaledrange",
    )
    parser.add_argument("-v", "--version", action="version", version="0.5.0")
    args = parser.parse_args()

    # Calculate scaled range [m/kton^(1/3)]
    rs = args.rangem / args.yieldkton ** (1 / 3)

    if 25 <= rs <= 250:
        print(
            "Scaled range = {0:.1f} m/\N{CUBE ROOT}ktonne in model range "
            "(25 to 250)".format(rs)
        )
    else:
        print(
            "Scaled range = {0:.1f} m/\N{CUBE ROOT}ktonne out of model range "
            "(25 to 250)".format(rs)
        )

    if args.transitionscaledrange is not None:
        print(
            "Option chosen to turn on elastic propagation (1/r) at {0:.1f} "
            "m/\N{CUBE ROOT}ktonne".format(args.transitionscaledrange)
        )

    print(
        "Ground motion [min to max] ({0:d}% single prediction interval)".format(
            args.confidencepercent
        )
    )

    amin = amasmin(rs, args.confidencepercent) / args.yieldkton ** (1 / 3)
    amax = amasmax(rs, args.confidencepercent) / args.yieldkton ** (1 / 3)
    amean = amas(rs) / args.yieldkton ** (1 / 3)

    if args.transitionscaledrange is not None:
        amin = amasminext(
            rs, args.confidencepercent, args.transitionscaledrange
        ) / args.yieldkton ** (1 / 3)
        amax = amasmaxext(
            rs, args.confidencepercent, args.transitionscaledrange
        ) / args.yieldkton ** (1 / 3)
        amean = amasext(rs, args.transitionscaledrange) / args.yieldkton ** (1 / 3)

    print("Acceleration = {0:g} to {1:g} g (Mean = {2:g} g)".format(amin, amax, amean))
```

```python

    vmin = amvmin(rs, args.confidencepercent)
    vmax = amvmax(rs, args.confidencepercent)
    vmean = amv(rs)

    if args.transitionscaledrange is not None:
        vmin = amvminext(rs, args.confidencepercent, args.transitionscaledrange)
        vmax = amvmaxext(rs, args.confidencepercent, args.transitionscaledrange)
        vmean = amvext(rs, args.transitionscaledrange)

    print("Velocity = {0:g} to {1:g} m/s (Mean = {2:g} m/s)".format(vmin, vmax, vmean))


## Scaled acceleration models


def pb75wtas(rs):
    """
    Perret & Bass [1975] scaled acceleration [g*kton^(1/3)] in wet-tuff
    """
    asmean = 4.31e7 * rs ** -2.61
    return asmean


def pb75wtasmin(rs):
    """
    Perret & Bass [1975] scaled acceleration [g*kton^(1/3)] in wet-tuff
    minimum from prediction interval
    """
    asmin = (4.31e7 * rs ** -2.61) / (1.96 * 2.21)
    return asmin


def pb75wtasmax(rs):
    """
    Perret & Bass [1975] scaled acceleration [g*kton^(1/3)] in wet-tuff
    maximum from prediction interval
    """
    asmax = (4.31e7 * rs ** -2.61) * (1.96 * 2.21)
    return asmax


def amas(rs):
    """
    Aqueduct Mesa model (Ford, 2020) scaled acceleration [g*kton^(1/3)]
    """
    x = log(rs)
    las = 21.1098 - 3.75861 * x
    #     las = 20.8003 - 3.70645*x # outliers removed
    asmean = exp(las)
    return asmean


def amasmin(rs, cl):
    """
    Aqueduct Mesa model (Ford, 2020) scaled acceleration [g*kton^(1/3)]
    Minimum from prediction interval at cl %
    """
    x = log(rs)
    b = sqrt(1.57775 - 0.466731 * x + 0.0562978 * x ** 2)
    las = 21.1098 - 3.75861 * x

    if cl == 90:
        lasmin = las - 1.70532 * b
    elif cl == 95:
        lasmin = las - 2.05553 * b
    else:  # cl == 99
        lasmin = las - 2.77871 * b

    asmin = exp(lasmin)
    return asmin


def amasmax(rs, cl):
```

```
    """
    Aqueduct Mesa model (Ford, 2020) scaled acceleration [g*kton^(1/3)]
    maximum from prediction interval at cl %
    """
    x = log(rs)
    b = sqrt(1.57775 - 0.466731 * x + 0.0562978 * x ** 2)
    las = 21.1098 - 3.75861 * x

    if cl == 90:
        lasmax = las + 1.70532 * b
    elif cl == 95:
        lasmax = las + 2.05553 * b
    else:  # cl == 99
        lasmax = las + 2.77871 * b

    asmax = exp(lasmax)
    return asmax


def amasext(rs, trs):
    """
    Aqueduct Mesa model (Ford, 2020) scaled acceleration [g*kton^(1/3)]
    Extended model with elastic (1/r) propagation (start at given scaled range)
    """
    x = log(rs)
    tx = log(trs)

    if rs <= trs:
        las = 21.1098 - 3.75861 * x
    else:
        las = 21.1098 - 3.75861 * tx + tx - x

    asmean = exp(las)
    return asmean


def amasminext(rs, cl, trs):
    """
    Aqueduct Mesa model (Ford, 2020) scaled acceleration [g*kton^(1/3)]
    Minimum from prediction interval at cl %
    Extended model with elastic (1/r) propagation (start at given scaled range)
    """
    x = log(rs)
    tx = log(trs)

    if rs <= trs:
        las = 21.1098 - 3.75861 * x
        b = sqrt(1.57775 - 0.466731 * x + 0.0562978 * x ** 2)
        if cl == 90:
            lasmin = las - 1.70532 * b
        elif cl == 95:
            lasmin = las - 2.05553 * b
        else:  # cl == 99
            lasmin = las - 2.77871 * b
    else:
        las = 21.1098 - 3.75861 * tx
        b = sqrt(1.57775 - 0.466731 * tx + 0.0562978 * tx ** 2)
        if cl == 90:
            lasmin = las - 1.70532 * b + tx - x
        elif cl == 95:
            lasmin = las - 2.05553 * b + tx - x
        else:  # cl == 99
            lasmin = las - 2.77871 * b + tx - x

    asmin = exp(lasmin)
    return asmin


def amasmaxext(rs, cl, trs):
    """
    Aqueduct Mesa model (Ford, 2020) scaled acceleration [g*kton^(1/3)]
    Maximum from prediction interval at cl %
    Extended model with elastic (1/r) propagation (start at given scaled range)
    """
```

```python
    x = log(rs)
    tx = log(trs)

    if rs <= trs:
        las = 21.1098 - 3.75861 * x
        b = sqrt(1.57775 - 0.466731 * x + 0.0562978 * x ** 2)
        if cl == 90:
            lasmax = las + 1.70532 * b
        elif cl == 95:
            lasmax = las + 2.05553 * b
        else:  # cl == 99
            lasmax = las + 2.77871 * b
    else:
        las = 21.1098 - 3.75861 * tx
        b = sqrt(1.57775 - 0.466731 * tx + 0.0562978 * tx ** 2)
        if cl == 90:
            lasmax = las + 1.70532 * b + tx - x
        elif cl == 95:
            lasmax = las + 2.05553 * b + tx - x
        else:  # cl == 99
            lasmax = las + 2.77871 * b + tx - x

    asmax = exp(lasmax)
    return asmax


## Velocity models


def pb75wtv(rs):
    """
    Perret & Bass [1975] velocity [m/s] in wet-tuff
    """
    vmean = 6.61e3 * rs ** -1.56
    return vmean


def pb75wtvmin(rs):
    """
    Perret & Bass [1975] scaled acceleration [g*kton^(1/3)] in wet-tuff
    minimum from prediction interval
    """
    vmin = (6.61e3 * rs ** -1.56) / (1.96 * 1.56)
    return vmin


def pb75wtvmax(rs):
    """
    Perret & Bass [1975] scaled acceleration [g*kton^(1/3)] in wet-tuff
    Maximum from prediction interval
    """
    vmax = (6.61e3 * rs ** -1.56) * (1.96 * 1.56)
    return vmax


def amv(rs):
    """
    Fourney et al. [1994] derived scaled acceleration [g*kton^(1/3)]
    """
    x = log(rs)
    lv = 9.83696 - 1.88593 * x
    #       lv = 8.99046 - 1.7263*x # outliers removed
    vmean = exp(lv)
    return vmean


def amvmin(rs, cl):
    """
    Aqueduct Mesa model (Ford, 2020) velocity [m/s]
    Minimum from prediction interval at cl %
    """
    x = log(rs)
    b = sqrt(1.96386 - 0.527575 * x + 0.0601074 * x ** 2)
    lv = 9.83696 - 1.88593 * x
```

```

    if cl == 90:
        lvmin = lv - 1.69913 * b
    elif cl == 95:
        lvmin = lv - 2.04523 * b
    else:  # cl == 99
        lvmin = lv - 2.75639 * b

    vmin = exp(lvmin)
    return vmin


def amvmax(rs, cl):
    """
    Aqueduct Mesa model (Ford, 2020) velocity [m/s]
    Maximum from prediction interval at cl %
    """
    x = log(rs)
    b = sqrt(1.96386 - 0.527575 * x + 0.0601074 * x ** 2)
    lv = 9.83696 - 1.88593 * x

    if cl == 90:
        lvmax = lv + 1.69913 * b
    elif cl == 95:
        lvmax = lv + 2.04523 * b
    else:  # cl == 99
        lvmax = lv + 2.75639 * b

    vmax = exp(lvmax)
    return vmax


def amvext(rs, trs):
    """
    Fourney et al. [1994] derived scaled acceleration [g*kton^(1/3)]
    """
    x = log(rs)
    tx = log(trs)

    if rs <= trs:
        lv = 9.83696 - 1.88593 * x
    else:
        lv = 9.83696 - 1.88593 * tx + tx - x

    vmean = exp(lv)
    return vmean


def amvminext(rs, cl, trs):
    """
    Aqueduct Mesa model (Ford, 2020) velocity [m/s]
    minimum from prediction interval at cl %
    """
    x = log(rs)
    tx = log(trs)

    if rs <= trs:
        lv = 9.83696 - 1.88593 * x
        b = sqrt(1.96386 - 0.527575 * x + 0.0601074 * x ** 2)
        if cl == 90:
            lvmin = lv - 1.69913 * b
        elif cl == 95:
            lvmin = lv - 2.04523 * b
        else:  # cl == 99
            lvmin = lv - 2.75639 * b
    else:
        lv = 9.83696 - 1.88593 * tx
        b = sqrt(1.96386 - 0.527575 * tx + 0.0601074 * tx ** 2)
        if cl == 90:
            lvmin = lv - 1.69913 * b + tx - x
        elif cl == 95:
            lvmin = lv - 2.04523 * b + tx - x
        else:  # cl == 99
            lvmin = lv - 2.75639 * b + tx - x
```

```python

    vmin = exp(lvmin)
    return vmin


def amvmaxext(rs, cl, trs):
    """
    Aqueduct Mesa model (Ford, 2020) velocity [m/s]
    Maximum from prediction interval at cl %
    """
    x = log(rs)
    tx = log(trs)

    if rs <= trs:
        lv = 9.83696 - 1.88593 * x
        b = sqrt(1.96386 - 0.527575 * x + 0.0601074 * x ** 2)
        if cl == 90:
            lvmax = lv + 1.69913 * b
        elif cl == 95:
            lvmax = lv + 2.04523 * b
        else:  # cl == 99
            lvmax = lv + 2.75639 * b
    else:
        lv = 9.83696 - 1.88593 * tx
        b = sqrt(1.96386 - 0.527575 * tx + 0.0601074 * tx ** 2)
        if cl == 90:
            lvmax = lv + 1.69913 * b + tx - x
        elif cl == 95:
            lvmax = lv + 2.04523 * b + tx - x
        else:  # cl == 99
            lvmax = lv + 2.75639 * b + tx - x

    vmax = exp(lvmax)
    return vmax


if __name__ == "__main__":
    main()
```