Obtención de la masa modela equivalente

|  |  |
| --- | --- |
|  |  |
|  |  |

Masas modal equivalente sobre longitud de vórtice

|  |  |
| --- | --- |
|  |  |
|  |  |

Constantes

|  |  |
| --- | --- |
| Densidad del aire |  |
| Buffeting | |
| Intensidad de turbulencia |  |
| Escala de longitud promedio |  |
| Parámetro de Kaimal |  |
| Parámetro de co-espectro |  |
| Velocidad |  |
| Desprendimiento de vórtices | |
| Número de Strouhal |  |
| Raíz adimensional media cuadrada del coeficiente de levante |  |
| Parámetro de banda de ancha del espectro de carga adimensional |  |
| Parámetro para auto limite característico |  |
| Longitud de escala de coherencia adimensional |  |
| Valor de referencia para la velocidad dependiente del coeficiente de amortiguamiento |  |

|  |  |
| --- | --- |
| Propiedades de la estructura | |
| Frecuencias de los primeros dos modos |  |
| Amortiguamiento para ambos modos |  |
| Masa modal para los primeros dos modos |  |

Buffeting

**Función de densidad espectral de Kaimal**

Regresa la función de densidad de auto espectro de Kaimal para las direcciones n=u(i=1) y n=w(i=2) (ecuación 1.32 de la tesis P.23).

|  |
| --- |
|  |

Parámetros de entrada

|  |  |
| --- | --- |
| Frecuencias del sistema |  |
| Parámetros de Kaimal |  |
| Escala de longitud promedio |  |
| Velocidad del viento |  |

Algoritmo

|  |
| --- |
| function [ S\_n\_red ] = KaimalSpectralDensity ( omega , A\_n , xL\_n , V)  S\_n\_red = zeros (2 , length ( omega )) ;  for i = 1:2  S\_n\_red (i ,:) = ( A\_n (i)\* xL\_n ( i)) ./( V \*(1+1.5\* A\_n ( i) .\* omega \*( xL\_n (i)/V )).^(5/3) );  end  end |

**Co-espectro**

Regresa el coespectro normalizado de la turbulencia del viento para las direcciones n=u(i=1) y n=w(i=2) (ecuación 1.33 de la tesis P.23).

|  |
| --- |
|  |

Parámetros de entrada

|  |  |
| --- | --- |
| Frecuencias del sistema |  |
| Longitud entre los puntos a lo largo del puente |  |
| Parametros de co-espectro |  |
| Velocidad del viento |  |

Algoritmo

|  |
| --- |
| function [ Co\_nn ] = Co\_spectrum ( omega ,Dx , c\_nx ,V)  Co\_nn = zeros (2 , length ( omega )) ;  for i = 1:2  Co\_nn (i ,:) = exp (- c\_nx (i ) \* ( omega \* Dx ) /(2 \* pi \* V));  end  end |

**Coeficiente de amortiguamiento aerodinámico debido al buffeting**

Regresa el coeficiente de amortiguamiento aerodinámico debido al buffeting (ecuación 2.2 y 2.3 de la tesis P.50)

|  |
| --- |
|  |

Parámetros de entrada

|  |  |
| --- | --- |
| Densidad del viento |  |
| Velocidad del viento |  |
| Ancho del puente |  |
| Peralte del puente |  |
| Coeficiente de arrastre |  |
| Coeficiente de levante |  |
| Las formas modales en las tres direcciones |  |
| Frecuencias de la estructura |  |
| Masa modal equivalente |  |
| Longitud total del puente |  |
| Dirección de análisis |  |
| Integral de la forma modal en la dirección a evaluar | ó |

Algoritmo

|  |
| --- |
| function [ zeta\_ae ] = AerodynamicDampingCoefficient ( rho ,V ,B ,D ,Cd , Cdl , Phi , omega\_e , m\_e , L\_el ,dir , IntPhi\_e )  Int = 0;  n = length ( L\_el );  if dir == 1  for i = 1: n  Int = Int + Phi (i) ^2\* D(i )\* Cd ( i)\* L\_el (i );  end  elseif dir == 2  for i = 1: n  Int = Int + Phi (i) ^2\*( B\* Cdl (i )+D( i)\* Cd (i))\* L\_el ( i);  end  else  disp ( ’ error ’)  end  zeta\_ae = -( rho \*V) /(2\* omega\_e \* m\_e ) \* Int / IntPhi\_e ;  end |

**Función de respuesta de frecuencias**

Regresa respuesta de frecuencia para el modo deseado (ecuación 1.62 de la tesis P.29)

|  |
| --- |
|  |

Parámetros de entrada

|  |  |
| --- | --- |
| Frecuencia del sistema |  |
| Densidad del viento |  |
| Coeficiente de amortiguamiento de la estructura |  |
| Velocidad del viento |  |
| Ancho del puente |  |
| Peralte del puente |  |
| Coeficiente de arrastre |  |
| Coeficiente de levante |  |
| Longitud del puente |  |
| Dirección de análisis |  |
|  |  |
| Las formas modales en las tres direcciones |  |
| Frecuencias de la estructura |  |
| Masa modal equivalente |  |
| Longitud total del puente |  |
| Integral de la forma modal en la dirección a evaluar | ó |

Algoritmo

|  |
| --- |
| function [ abs\_H ] = FrequencyResponse ( omega , rho , zeta\_e ,V ,B ,D ,Cd , Cdl , L\_el , dir , Phi ,omega\_e , m\_e , IntPhi\_e )  zeta\_ae = AerodynamicDampingCoefficient ( rho ,V ,B ,D ,Cd , Cdl , Phi , omega\_e , m\_e , L\_el ,dir , IntPhi\_e );  abs\_H = abs (1 - ( omega ./ omega\_e ) .^2 + 2\*1 i \*( zeta\_e - zeta\_ae ) \*(( omega ./ omega\_e ))) .^( -1) ;  end |

**Función de incremento de variables**

Dentro de una matriz de datos, divide en una cantidad definida de variables dichos datos

Parámetros de entrada

|  |  |
| --- | --- |
| Variable a subdividir |  |
| Cantidad de divisiones |  |

Algoritmo

|  |
| --- |
| function [ Varnew ] = Prolong ( Var , div )  n = length ( Var );  Varnew = zeros (0 ,(n -1) \* div +1) ;  Varnew (1) = Var (1) ;  for i = 0:n -2  delta = ( Var (i +2) - Var (i +1) )/ div ;  for j = 1: div  Varnew ( div \*i +j +1) = Varnew ( div \*i+j ) + delta ;  end  end  end |

**Función de expansión de variables**

Dentro de una matriz de datos, divide en una cantidad definida de variables dichos datos

Parámetros de entrada

|  |  |
| --- | --- |
| Variable a subdividir |  |
| Cantidad de divisiones |  |

Algoritmo

|  |
| --- |
| function [ Varnew ] = Expand ( Var , div )  n = length ( Var );  Varnew = zeros (0 ,(n -1) \* div +1) ;  Varnew (2: div ) = ( Var (1) + Var (2) \*0.5) / div ;  Varnew (1) = 0.5\* Varnew (2) ;  for i = 1:n -3  delta = ( Var (i +1) + Var (i +2) ) /(2\* div );  Varnew ( div \*i +1: div \*i+ div ) = delta ;  Varnew ( div \*i + div ) = ( delta + ( Var (i +2) + Var (i +3) ) /(2\* div )) /2;  end  Varnew ((n -2) \* div +1:( n -1) \* div ) = ( Var (n -1) \*0.5+ Var (n)) / div ;  Varnew ((n -1) \* div +1) = 0.5\* Varnew ((n -1) \* div ) ;  Varnew ( div \*(n -3) + div ) = ( delta + ( Var (i +2) +2\* Var (i +3) ) /(2\* div )) /2;  end |

**Función de aceptación**

Regresa la función de aceptancia normalizada del modo deseado (ecuación 1.70 de la tesis P.32)

|  |
| --- |
|  |

Parámetros de entrada

|  |  |
| --- | --- |
| Frecuencia del sistema |  |
| Numero de divisiones |  |
| Velocidad del viento |  |
| Ancho del puente |  |
| Peralte del puente |  |
| Coeficiente de arrastre |  |
| Coeficiente de levante |  |
| Coeficiente de levante |  |
| Longitud del puente |  |
| Parametros de co-espectro |  |
| Parámetro de Kaimal |  |
| Escala de longitud promedio |  |
| Intensidad de turbulencia |  |
| Dirección de análisis |  |
| Las formas modales en las tres direcciones |  |
| Integral de la forma modal en la dirección a evaluar | ó |

Algoritmo

|  |
| --- |
| function [ Jnorm ] = JointAcceptance ( omega , div ,V ,B ,D ,Cd ,Cl , Cdl , L\_el , c\_nx , A\_n , xL\_n ,  I\_n ,dir , Phi , IntPhi\_e )  % Returns the normalized joint acceptance function of mode Phi  % Defining variables :  Nomega = length ( omega );  J = zeros (1 , Nomega );  xi = 0;  % The reduced Kaimal spectral density :  S\_n\_red = KaimalSpectralDensity ( omega , A\_n , xL\_n , V);  % Linear prolongation / expandation of input variables :  D = Prolong (D , div );  Cd = Prolong ( Cd , div );  Cdl = Prolong ( Cdl , div ) ;  Phi = Prolong ( Phi , div ) ;  L\_el = Expand ( L\_el , div );  n = length ( L\_el );  % Integration :  for i = 1: n  Di = D(i );  Cdi = Cd (i);  Cdli = Cdl (i);  L\_eli = L\_el (i);  xi = xi + L\_eli ;  Phii = Phi (i);  f  xj = 0;  for j = 1: n  Dj = D(j );  Cdj = Cd (j);  Cdlj = Cdl (j);  L\_elj = L\_el (j);  xj = xj + L\_elj ;  Phij = Phi (j);  Dx = abs (xi - xj );  Coij = Co\_spectrum ( omega ,Dx , c\_nx , V);  if dir == 1 % mode in y - direction  J = J + Phii \* Phij .\* ( (((2\* I\_n (1) )/ B) ^2\* Di \* Cdi \* Dj \* Cdj ) .\* Coij  (1 ,:) .\* S\_n\_red (1 ,:) + ( Cl \* I\_n (2) ) ^2 .\* Coij (2 ,:) .\* S\_n\_red  (2 ,:) ) \* L\_eli \* L\_elj ;  elseif dir == 2 % mode in z - direction  J = J + Phii \* Phij .\* ( (2\* Cl \* I\_n (1) ) ^2 .\* Coij (1 ,:) .\* S\_n\_red  (1 ,:) + (( I\_n (2) ^2\*( Cdli +( Di \* Cdi )/B) \*( Cdlj +( Dj \* Cdj )/ B)) .\*  Coij (2 ,:) .\* S\_n\_red (2 ,:) )) \* L\_eli \* L\_elj ;  else  disp ( ’ error ’)  end  end  end  % Normalizing the joint acceptance function :  Jnorm = J /( IntPhi\_e ) ^2;  end |

**Desviación estandar**

Regresa la desviación estandar del modo deseado (ecuación 1.71 de la tesis P.32)

|  |
| --- |
|  |

Parámetros de entrada

|  |  |
| --- | --- |
| Frecuencia del sistema |  |
| Numero de divisiones |  |
| Densidad del viento |  |
| Amortiguamiento de la estructura |  |
| Velocidad del viento |  |
| Ancho del puente |  |
| Peralte del puente |  |
| Coeficiente de arrastre |  |
| Coeficiente de levante |  |
| Coeficiente de levante |  |
| Longitud del puente |  |
| Parametros de co-espectro |  |
| Parámetro de Kaimal |  |
| Escala de longitud promedio |  |
| Intensidad de turbulencia |  |
| Dirección de análisis |  |
| Frecuencias de la estructura |  |
| Masa modal equivalente |  |
| Las formas modales en las tres direcciones |  |
| Integral de la forma modal en la dirección a evaluar | ó |

Algoritmo

|  |
| --- |
| function sigma\_r = StandardDeviation ( omega , div , rho , zeta\_e ,V ,B ,D ,Cd ,Cl , Cdl , L\_el ,  c\_nx , A\_n , xL\_n , I\_n ,dir , Phi , omega\_e , m\_e , IntPhi\_e )  % Calculates standard deviation of response along x - axis for mode Phi  Nomega = length ( omega );  domega = ( omega ( Nomega ) - omega (1) ) /( Nomega -1) ;  Int = 0;  % Frequency response function :  abs\_H = FrequencyResponse ( omega , rho , zeta\_e ,V ,B ,D ,Cd , Cdl , L\_el , dir , Phi , omega\_e ,  m\_e , IntPhi\_e );  % Normalized joint acceptance function :  Jnorm = JointAcceptance ( omega , div ,V ,B ,D ,Cd ,Cl , Cdl , L\_el , c\_nx , A\_n , xL\_n , I\_n , dir ,  Phi , IntPhi\_e );  % Integration :  for i = 1: Nomega  Int = Int + abs\_H (i ) ^2\* Jnorm ( i);  end  % Standard deviation of the response :  sigma\_r = abs ( Phi ) \* (( rho \*V ^2\* B) /(2\* m\_e \* omega\_e ^2) \* sqrt ( Int \* domega )) ;  end |

**Gráfica de la desviación estándar**

Gráfica la desviación estándar de los modo

Parámetros necesarios para que corra

|  |  |
| --- | --- |
| Matriz de elementos |  |
| Numero de modos (dirección y y z)s |  |
| Densidad del viento |  |
| Amortiguamiento de la estructura |  |
| Velocidad del viento |  |
| Ancho del puente |  |
| Peralte del puente |  |
| Coeficiente de arrastre |  |
| Coeficiente de levante |  |
| Coeficiente de levante |  |
| Longitud del puente |  |
| Parametros de co-espectro |  |
| Parámetro de Kaimal |  |
| Escala de longitud promedio |  |
| Intensidad de turbulencia |  |
| Dirección de análisis |  |
| Frecuencias de la estructura |  |
| Masa modal equivalente |  |
| Las formas modales en las tres direcciones |  |
| Integral de la forma modal en la dirección a evaluar | ó |

Algoritmo

|  |
| --- |
| clear all  close all  Input  % Choose integration accuracy :  omega = linspace (0 ,20 ,2000) ;  div = 4;  n = length ( L\_el );  sigma\_r = zeros ( n\_mod ,n );  for i = 1: n\_mod  sigma\_r (i ,:) = StandardDeviation ( omega , div , rho , zeta\_e ,V ,B ,D ,Cd ,Cl , Cdl , L\_el ,  c\_nx , A\_n , xL\_n , I\_n ,i , Phi (i ,:) , omega\_e (i) , m\_e (i ) , IntPhi\_e (i)) ;  plot ( x\_vect , sigma\_r (i ,:) ,’ LineWidth ’ ,2)  hold all  end  grid  xlabel ( ’x [m] ’)  ylabel ( ’\ sigma\_ {r }( x) [m] ’)  legend ( ’\ sigma\_ { r\_ { y }}( x ) ’,’\ sigma\_ { r\_ {z }}( x) ’) |

**Desviación estandar divido por fi**

Regresa la desviación estandar del modo deseado divido por fi (ecuación 1.71 de la tesis P.32)

|  |
| --- |
|  |

Parámetros de entrada

|  |  |
| --- | --- |
| Frecuencia del sistema |  |
| Numero de divisiones |  |
| Densidad del viento |  |
| Amortiguamiento de la estructura |  |
| Velocidad del viento |  |
| Ancho del puente |  |
| Peralte del puente |  |
| Coeficiente de arrastre |  |
| Coeficiente de levante |  |
| Coeficiente de levante |  |
| Longitud del puente |  |
| Parametros de co-espectro |  |
| Parámetro de Kaimal |  |
| Escala de longitud promedio |  |
| Intensidad de turbulencia |  |
| Dirección de análisis |  |
| Frecuencias de la estructura |  |
| Masa modal equivalente |  |
| Las formas modales en las tres direcciones |  |
| Integral de la forma modal en la dirección a evaluar | ó |

Algoritmo

|  |
| --- |
| function S\_r\_dphi = ResponseSpectrum ( omega , div , rho , zeta\_e ,V ,B ,D , Cd ,Cl , Cdl , L\_el ,  c\_nx , A\_n , xL\_n , I\_n ,dir , Phi , omega\_e , m\_e , IntPhi\_e )  % Calculates response spectrum divided on Phi ( x\_r )  % Frequency response function :  abs\_H = FrequencyResponse ( omega , rho , zeta\_e ,V ,B ,D ,Cd , Cdl , L\_el , dir , Phi , omega\_e ,  m\_e , IntPhi\_e );  % Normalized joint acceptance function :  Jnorm = JointAcceptance ( omega , div ,V ,B ,D ,Cd ,Cl , Cdl , L\_el , c\_nx , A\_n , xL\_n , I\_n , dir ,  Phi , IntPhi\_e );  % Response spectrum :  S\_r\_dphi = (( rho \*V ^2\* B) /(2\* m\_e \* omega\_e ^2) ) ^2 \* abs\_H .^2.\* Jnorm ;  end |

**Gráfica de la simulación en el dominio del tiempo**

Gráfica la desviación estándar de los modo

Parámetros necesarios para que corra

|  |  |
| --- | --- |
| Matriz de elementos |  |
| Numero de modos (dirección y y z)s |  |
| Densidad del viento |  |
| Amortiguamiento de la estructura |  |
| Velocidad del viento |  |
| Ancho del puente |  |
| Peralte del puente |  |
| Coeficiente de arrastre |  |
| Coeficiente de levante |  |
| Coeficiente de levante |  |
| Longitud del puente |  |
| Parametros de co-espectro |  |
| Parámetro de Kaimal |  |
| Escala de longitud promedio |  |
| Intensidad de turbulencia |  |
| Dirección de análisis |  |
| Frecuencias de la estructura |  |
| Masa modal equivalente |  |
| Las formas modales en las tres direcciones |  |
| Integral de la forma modal en la dirección a evaluar | ó |

Algoritmo

|  |
| --- |
| % TIME DOMAIN SIMULATIONS OF BUFFETING RESPONSE  clear all  close all  Input  % Fill in for sigma\_r :  sigma\_r = [0.139 0.076];  % Choose integration accuracy :  omega = linspace (0 ,20 ,2000) ;  div = 4;  N\_omega = length ( omega );  domega = ( omega ( N\_omega ) - omega (1) ) / N\_omega ;  % Simulate over 10 minutes ’ periodes :  t = linspace (0 ,600 ,600) ;  n\_t = length (t) ;  % Define variables :  S\_r\_dphi = zeros ( n\_mod , N\_omega );  S\_r = zeros ( n\_mod , N\_omega );  r = zeros ( n\_mod , n\_t );  % Response at the tip of the cantilever :  for i = 1: n\_mod  Int =0;  h  S\_r\_dphi (i ,:) = ResponseSpectrum ( omega , div , rho , zeta\_e ,V ,B ,D , Cd ,Cl , Cdl , L\_el ,  c\_nx , A\_n , xL\_n , I\_n ,i , Phi (i ,:) , omega\_e (i) , m\_e (i ) , IntPhi\_e (i)) ;  S\_r (i ,:) = S\_r\_dphi (i ,:) \* Phi (i ,1) ^2;  % Peak factor :  maxv = max ( abs (r(i ,:) ) );  kp = maxv / sigma\_r ( i)  end  % Plot the response of time :  subplot (2 ,1 ,1)  plot (t , r (1 ,:) )  grid  xlabel ( ’T [s] ’)  ylabel ( ’ r\_y [m] ’)  hold all  subplot (2 ,1 ,2)  plot (t , r (2 ,:) )  grid  xlabel ( ’T [s] ’)  ylabel ( ’ r\_z [m] ’)  % 3 D PLOT OF RESPONSE MODE 1  clear all  close all  Input  n = length ( L\_el );  zoom = 30;  rmax = 0.44;  hold all  % Pillar initially :  xpillar0 = zeros (1 ,10) ;  ypillar0 = zeros (1 ,10) ;  zpillar0 = linspace (0 ,30 ,10) ;  plot3 ( xpillar0 , ypillar0 , zpillar0 , ’ -b ’, ’ Linewidth ’ ,2)  % Girder initially :  xgirder0 = x\_vect ;  ygirder0 = zeros (1 , n);  zgirder0 = ones (1 , n) \*30;  plot3 ( xgirder0 , ygirder0 , zgirder0 , ’ -k ’, ’ Linewidth ’ ,1)  % Girder new :  xgirder = x\_vect ;  ygirder = zoom \* rmax \* Phi (1 ,:) ;  zgirder = zgirder0 ;  plot3 ( xgirder , ygirder , zgirder , ’-b ’ ,’ Linewidth ’ ,2)  xlabel ( ’x ’)  ylabel ( ’y ’)  zlabel ( ’z ’)  grid on  axis ([ -100 100 -50 50 0 40]) |

**Desprendimiento de vórtices**

**Coeficiente de amortiguamiento aerodinámico**

Regresa la desviación estandar del modo deseado divido por fi (ecuación 1.88 de la tesis P.35)

|  |
| --- |
|  |

Parámetros de entrada

|  |  |
| --- | --- |
| Ancho del puente |  |
| Peralte del puente |  |
| Densidad del viento |  |
| Valor asociado con limite característico del vórtice |  |
| Coeficiente de velocidad de amortiguamiento |  |
| Masa modal equivalente |  |
| Desviación estándar |  |
| Integral a lo largo del vórtice |  |
| Integral a lo largo de la estructura |  |

Algoritmo

|  |
| --- |
| function [ zeta\_ae ] = AerodynamicDampingCoefficientV (B ,D , rho , a\_z , K\_az , m\_e , sigma\_r ,  IntPhi\_e , IntPhi\_Lexp )  zeta\_ae = (( rho \*B ^2\* K\_az ) /(4\* m\_e ) ) .\*(1 -( sigma\_r ./( a\_z \*D)) .^2) \*( IntPhi\_e /  IntPhi\_Lexp );  end |

**Co-espectro normalizado del desprendimiento de vórtice**

Regresa el co-espectro normalizado del desprendimiento de vórtice en la dirección z (ecuación 1.51 de la tesis P.27)

|  |
| --- |
|  |

Parámetros de entrada

|  |  |
| --- | --- |
| Peralte de la sección |  |
| Longitud de escala adimensional de coherencia |  |
| Separación a lo largo del puente |  |

Algoritmo

|  |
| --- |
| function [ Co\_qz ] = Co\_spectrumV (D , lambda , Dx )  Co\_qz = cos ((2\* Dx ) ./(3\* lambda \*D)) .\* exp ( -(( Dx ./(3\* lambda \*D)) .^2) ) ;  end |

**Espectro de carga modal**

Regresa el espectro de carga modal para una velocidad definida (ecuación 2.17 de la tesis P.58)

|  |
| --- |
|  |

Parámetros de entrada

|  |  |
| --- | --- |
| Frecuencia del sistema |  |
| Densidad del viento |  |
| Ancho del puente |  |
| Peralte del puente |  |
| Número de Strouhal |  |
| Raíz adimensional media cuadrada del coeficiente de levante |  |
| Parámetro de banda de ancha del espectro de carga adimensional |  |
| Longitud de escala adimensional de coherencia |  |
| Integral a lo largo del vórtice |  |
| Velocidad media del viento |  |

Algoritmo

|  |
| --- |
| function S\_Q = LoadSpectrumV ( omega , rho ,B ,D ,St , sigma\_qz , b\_z , lambda\_z , IntPhi\_Lexp ,V)  omega\_s = (2\* pi\* St \*V)/D ;  S\_Q = 2\* lambda\_z \* D \* (((0.5\* rho \*V ^2\* B\* sigma\_qz ) ^2) /( sqrt (pi)\* omega\_s \* b\_z )) \* exp  ( -((1 - omega ./ omega\_s ) ./ b\_z ) .^2) .\* IntPhi\_Lexp ;  end |

**Desviación estándar**

Regresa el espectro de carga modal para una velocidad definida (ecuación 1.91-1.94 de la tesis P.35 y P.36)

|  |
| --- |
|  |

Parámetros de entrada

|  |  |
| --- | --- |
| Densidad del viento |  |
| Ancho del puente |  |
| Peralte del puente |  |
| Número de Strouhal |  |
| Raíz adimensional media cuadrada del coeficiente de levante |  |
| Parámetro de banda de ancha del espectro de carga adimensional |  |
| Longitud de escala adimensional de coherencia |  |
| Coeficiente de velocidad de amortiguamiento |  |
| Amortiguamiento de la estructura |  |
| Amortiguamiento de la estructura |  |
| Masa modal equivalente |  |
| Forma modal |  |
| Integral a lo largo del puente |  |
| Integral a lo largo del vórtice |  |
| Velocidad media del viento |  |

Algoritmo

|  |
| --- |
| function sigma\_r = StandardDeviationV ( rho ,B ,D ,St , sigma\_qz , a\_z , b\_z , lambda\_z , K\_az ,  zeta\_e , omega\_e , m\_e , Phi , IntPhi\_e , IntPhi\_Lexp , V)  % Resonant velocity :  V\_R = ( D\* omega\_e ) /(2\* pi\* St );  % Temporary variables :  zeta\_hat = ((4\* m\_e \* zeta\_e ) /( rho \*B ^2\* K\_az )) \*( IntPhi\_e / IntPhi\_Lexp );  beta\_hat = Phi /(2^(5/2) \* pi ^(7/4) ) \* (( rho \*D ^3\* lambda\_z ) /( m\_e \* b\_z \* K\_az \* IntPhi\_e ) )  ^(1/2) \* ( sigma\_qz /( St ^2\* a\_z ) ) \* (V / V\_R ) ^(3/2) \* exp ( -(1/2) \*((1 -( V\_R /V)) / b\_z )  ^2) ;  % Solves the fourth order polynomial :  sigma\_hat = ((1 - zeta\_hat ) /2 + (((1 - zeta\_hat ) /2) ^2 + beta\_hat ^2) ^(1/2) ) ^(1/2) ;  % Standard deviation of displacement :  sigma\_r = sigma\_hat \* a\_z \*D;  end |

**Gráfica de la desviación estándar de la respuesta del desprendimiento de vórtices**

Parámetros para que funcione

|  |  |
| --- | --- |
| Densidad del viento |  |
| Ancho del puente |  |
| Peralte del puente |  |
| Número de Strouhal |  |
| Raíz adimensional media cuadrada del coeficiente de levante |  |
| Parámetro de banda de ancha del espectro de carga adimensional |  |
| Longitud de escala adimensional de coherencia |  |
| Coeficiente de velocidad de amortiguamiento |  |
| Amortiguamiento de la estructura |  |
| Frecuencia de la estructura |  |
| Masa modal equivalente |  |
| Forma modal |  |
| Integral a lo largo del puente |  |
| Integral a lo largo del vórtice |  |
| Velocidad media del viento |  |

Algoritmo

|  |
| --- |
| % PLOT STANDARD DEVIATION OF VORTEX SHEDDING RESPONSE  clear all  close all  Input  omega\_e = omega\_e (2) ;  m\_e = m\_e (2) ;  IntPhi\_e = IntPhi\_e (2) ;  n = length ( L\_el );  % Plots the standard deviation for different velocities :  V = linspace (0 ,20 ,10000) ;  n\_V = length (V) ;  Phii = Phi (2 ,1) ; % Mode 2 evaluated in x = -98.7 m  sigma\_r = zeros (1 , n\_V );  for i = 1: n\_V  Vi = V(i );  sigma\_r (i) = StandardDeviationV ( rho ,B , D\_Vortex , St , sigma\_qz , a\_z , b\_z , lambda\_z ,  K\_az , zeta\_e , omega\_e , m\_e , Phii , IntPhi\_e , IntPhi\_Lexp , Vi );  end  subplot (2 ,1 ,1)  plot (V , sigma\_r , ’ Linewidth ’ ,2)  ylabel ( ’\ sigma\_ { r\_ { z }}( x = -98.7) [m] ’)  xlabel ( ’V [m/ s] ’)  grid  axis ([0 20 0 0.15])  % Plots the standard deviation along the girder at resonant wind velocity :  V = ( D\_Vortex \* omega\_e ) /(2\* pi\* St );  sigma\_r = zeros (1 , n);  for i = 1: n  Phii = Phi (2 , i );  sigma\_r (i) = StandardDeviationV ( rho ,B , D\_Vortex , St , sigma\_qz , a\_z , b\_z , lambda\_z ,  K\_az , zeta\_e , omega\_e , m\_e , Phii , IntPhi\_e , IntPhi\_Lexp , V);  end  subplot (2 ,1 ,2)  plot ( x\_vect , sigma\_r , ’ Linewidth ’ ,2)  ylabel ( ’\ sigma\_ { r\_ { z }}( x ) [ m] ’)  xlabel ( ’x [m] ’)  grid  axis ([ -100 100 0 0.15]) |

**Simulación en el dominio del tiempo de la respuesta del desprendimiento de vórtices**

Regresa el espectro de carga modal para una velocidad definida (ecuación 1.91-1.94 de la tesis P.35 y P.36)

Parámetros de entrada

|  |  |
| --- | --- |
| Densidad del viento |  |
| Ancho del puente |  |
| Peralte del puente |  |
| Número de Strouhal |  |
| Raíz adimensional media cuadrada del coeficiente de levante |  |
| Parámetro de banda de ancha del espectro de carga adimensional |  |
| Longitud de escala adimensional de coherencia |  |
| Coeficiente de velocidad de amortiguamiento |  |
| Amortiguamiento de la estructura |  |
| Amortiguamiento de la estructura |  |
| Masa modal equivalente |  |
| Forma modal |  |
| Integral a lo largo del puente |  |
| Integral a lo largo del vórtice |  |
| Velocidad media del viento |  |

Algoritmo

|  |
| --- |
| % TIME DOMAIN SIMULATION OF VORTEX SHEDDING RESPONSE  clear all  close all  Input  omega = linspace (0 ,20 ,2000) ;  N\_omega = length ( omega );  domega = ( omega ( N\_omega ) - omega (1) ) / N\_omega ;  t = linspace (0 ,600 ,600) ;  n\_t = length (t) ;  % Define the mode for which vortex shedding may be a problem :  Phii = Phi (2 ,1) ; % Largest displacement --> left tip of the cantilever  omega\_e = omega\_e (2) ;  m\_e = m\_e (2) ;  IntPhi\_e = IntPhi\_e (2) ;  % Fill in for sigma\_r :  sigma\_r = 0.130;  % Resonant velocity :  V = ( D\_Vortex \* omega\_e ) /(2\* pi\* St );  % Aerodynamic damping coefficient :  zeta\_ae = AerodynamicDampingCoefficientV (B , D\_Vortex , rho , a\_z , K\_az , m\_e , sigma\_r ,  IntPhi\_e , IntPhi\_Lexp );  % Frequency response function :  abs\_H = abs (1 - ( omega ./ omega\_e ) .^2 + 2\*1 i \*( zeta\_e - zeta\_ae ) \*(( omega ./ omega\_e )))  .^( -1) ;  k  % Load spectrum :  S\_Q = LoadSpectrumV ( omega , rho ,B , D\_Vortex , St , sigma\_qz , b\_z , lambda\_z , IntPhi\_Lexp ,V) ;  % Response spectrum :  S\_r = (( Phii ^2\* abs\_H .^2) ./(( omega\_e ^2\* m\_e \* IntPhi\_e ) .^2) ) .\* S\_Q ;  r = zeros (1 , n\_t );  int =0;  % Response at the tip of the cantilever :  for i = 1: N\_omega  r = r + sqrt (2\* S\_r (i)\* domega ) .\* cos ( omega (i )\*t - rand (1) \*2\* pi);  int = int + S\_r (i);  end  % Max value :  maxz = max ( abs (r) );  kpz = maxz / sigma\_r  % Plot the response of time :  plot (t , r)  grid  xlabel ( ’T [s] ’)  ylabel ( ’ r\_z [m] ’)  % 3 D PLOT OF RESPONSE MODE 2  clear all  close all  Input  n = length ( L\_el );  zoom = 30;  rmax = 0.33;  hold all  % Pillar initially :  xpillar0 = zeros (1 ,10) ;  ypillar0 = zeros (1 ,10) ;  zpillar0 = linspace (0 ,30 ,10) ;  plot3 ( xpillar0 , ypillar0 , zpillar0 , ’ -k ’, ’ Linewidth ’ ,1)  % Girder initially ;  xgirder0 = x\_vect ;  ygirder0 = zeros (1 , n);  zgirder0 = ones (1 , n) \*30;  plot3 ( xgirder0 , ygirder0 , zgirder0 , ’ -k ’, ’ Linewidth ’ ,1)  % Pillar new :  xpillar = zoom \* rmax  \*[0 ,0.00043 ,0.00349 ,0.00929 ,0.01761 ,0.02823 ,0.04092 ,0.05546 ,0.07163 ,0.10794];  ypillar = zeros (1 ,10) ;  zpillar = linspace (0 ,30 ,10) ;  plot3 ( xpillar , ypillar , zpillar , ’-b ’ ,’ Linewidth ’ ,2)  % Girder new :  xgirder = x\_vect + xpillar (10) ;  ygirder = zeros (1 , n);  zgirder = 30 + zoom \* rmax \* Phi (2 ,:) ;  plot3 ( xgirder , ygirder , zgirder , ’-b ’ ,’ Linewidth ’ ,2)  l  xlabel ( ’x ’)  ylabel ( ’y ’)  zlabel ( ’z ’)  grid on  axis ([ -100 100 -50 50 0 40]) |