

InterpolacionNumericaPro1 v1.0.0

Estimación estadístico-numérica de sólidos suspendidos totales mediante interpolación polinómica de Newton, regularizada en tramos fluviales

Año: 2026

Autor: JOSE ANGEL GASPAR GENICO

Titular de los derechos: JOSE ANGEL GASPAR GENICO

Tipo de obra: Programa de Cómputo

Resumen técnico: Programa de cómputo orientado a la estimación estadístico-numérica de Sólidos Suspendidos Totales (SST) en ríos con muestreo irregular, para apoyar decisiones operativas en plantas potabilizadoras de agua potable para consumo humano.

CODIGO_FUENTE_INTERPOLACIONNUMERICAPRO1

CÓDIGO FUENTE DEL PROGRAMA DE COMPUTACIÓN INTERPOLACIONNUMERICAPRO1

- Tipo de obra: Programa de Computación
- Plataforma: Microsoft Excel + VBA
- Versión: V1.0.0
- Fecha 3 de junio de 2026
- Nota:

El presente documento contiene la totalidad del código fuente del software INTERPOLACIONNUMERICAPRO1, sin omisiones, para fines de inspección pericial ante el Instituto Nacional del Derecho de Autor.

ÍNDICE GENERAL

1.	Estructura general del proyecto	5
1.1	Descripción general del proyecto VBA.....	5
1.2.	Componentes del proyecto	5
1.3	Organización lógica del código	6
2.	Código de eventos del libro	6
2.1	Evento Workbook_Open	6
2.2	Evento Workbook_BeforeClose	7
2.3	Otros eventos del libro (si aplica).....	8
3.	Código de hojas de cálculo	8
3.1	Hoja 1.....	8
3.1.1	Eventos y procedimientos asociados	9
3.2	Hoja 4.....	9
3.2.1	Eventos y procedimientos asociados.....	9
3.3	Hoja 5.....	9
3.3.1	Eventos y procedimientos asociados.....	10
4.	Código de UserForms	10
4.1	UserForm frmInicio.....	10
4.1.1	Inicialización del formulario	11
4.1.2	Eventos de controles	12
4.1.3	Procedimientos internos.....	15
5.	Código de módulos estándar	16
5.1	modAlgoritmos	17
5.2	modAppCore	20
5.3	modCapturaDatos.....	24
5.4	modConfiguracion.....	24
5.5	modEstadistica	27

5.6 modInterpolacionNewton	36
5.7 modReportes	40
5.8 modSemiotica	45
5.9. modSistema	48
5.10 modValidaciones.....	50
6. FUNDAMENTO MATEMÁTICO INTEGRADO.....	53
6.1. Interpolación polinómica de Newton	53
6.2. Fundamento matemático coherente con modInterpolacionNewton	54
6.3. Enfoque en estimación de SST.....	54
7. CÓDIGO FUENTE NUMERADO.....	55
7.1. ThisWorkbook	55
7.2. Hoja1	56
7.3. frmInicio	56
7.4. Todos los módulos listados	75
7.5. Numeración consecutiva	75
7.6. Comentarios técnicos en español	76

1. Estructura general del proyecto

El programa de cómputo **InterpolacionNumericaPro1** se encuentra organizado como un proyecto VBA estructurado, desarrollado sobre la plataforma Microsoft Excel, en el cual Excel funge únicamente como entorno de ejecución. El sistema está diseñado bajo un enfoque modular, donde cada componente cumple una función específica dentro de la operación general del software, permitiendo un control claro del flujo, del procesamiento de datos y de la interacción con el usuario.

1.1 Descripción general del proyecto VBA

El proyecto VBA implementa la lógica computacional del sistema mediante procedimientos, funciones y eventos programados, responsables de la inicialización del entorno, la captura y validación de datos, el procesamiento matemático y la presentación de resultados. La interacción del usuario se gestiona principalmente a través de formularios, evitando la manipulación directa de celdas y reforzando el comportamiento del archivo como una aplicación de software autónoma.

1.2. Componentes del proyecto

El proyecto se compone de diversos elementos propios del entorno VBA, entre los que se incluyen: objetos del libro (ThisWorkbook), hojas de cálculo con código asociado, formularios de usuario (UserForms) y módulos estándar. Cada componente ha sido diseñado para cumplir una función específica, ya sea de control del sistema, interfaz de usuario, procesamiento de datos o ejecución de algoritmos matemáticos, manteniendo una separación clara de responsabilidades.

1.3 Organización lógica del código

La organización lógica del código responde a un enfoque de desarrollo estructurado, en el cual la lógica de control, los algoritmos matemáticos, las validaciones y los procesos auxiliares se encuentran distribuidos en módulos independientes. Esta organización facilita la trazabilidad del código, su mantenimiento y su revisión pericial, permitiendo identificar con claridad la función de cada bloque dentro del sistema.

2. Código de eventos del libro

Este apartado contiene los procedimientos asociados a los eventos del libro de Excel que permiten controlar el comportamiento general del sistema durante su ejecución. Dichos eventos gestionan acciones automáticas al momento de abrir y cerrar el archivo, asegurando la correcta inicialización del entorno, la preparación de la interfaz y el cierre controlado del programa, reforzando su funcionamiento como una aplicación de software y no como una hoja de cálculo convencional.

2.1 Evento Workbook_Open

El evento Workbook_Open se ejecuta automáticamente al abrir el archivo y tiene como finalidad inicializar el sistema, configurar el entorno de ejecución y preparar la interfaz de usuario. A través de este evento se controla el acceso al programa y se establecen las condiciones necesarias para su operación correcta desde el inicio.

```
Private Sub Workbook_Open()
```

```

' Ocultar Excel completamente
Application.Visible = False
Application.ScreenUpdating = False

' Mostrar interfaz principal
frmInicio.Show

End Sub
` `

```

2.2 Evento Workbook_BeforeClose

El evento Workbook_BeforeClose se ejecuta al momento de cerrar el archivo y permite realizar acciones de control previo al cierre del sistema, tales como liberación de recursos, validaciones finales o restauración del entorno. Este evento contribuye a un cierre ordenado del programa y a la integridad del funcionamiento general del software.

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)
```

```

' Restaurar Excel antes de cerrar
Application.Visible = True
Application.ScreenUpdating = True

```

```
End Sub
```

2.3 Otros eventos del libro (si aplica)

El libro no contiene otros eventos adicionales distintos a los de apertura y cierre, ya que el control principal del sistema se gestiona mediante formularios de usuario y módulos estándar, manteniendo la lógica de ejecución centralizada y organizada.

3. Código de hojas de cálculo

Las hojas de cálculo del proyecto forman parte de la estructura interna del sistema y se utilizan como soporte de datos y control, bajo la supervisión de código VBA. Aunque contienen información visible, su comportamiento y uso dentro del software se encuentran restringidos y controlados por programación, evitando la manipulación libre por parte del usuario y manteniendo la coherencia del sistema.

3.1 Hoja 1

La Hoja 1 se utiliza como componente de soporte de datos del sistema, conteniendo información estructurada necesaria para los procesos de cálculo y análisis. Su función se encuentra integrada al flujo del software y no constituye un elemento de interacción directa libre, ya que su uso está condicionado por la lógica implementada en VBA.

3.1.1 Eventos y procedimientos asociados

La Hoja 1 no contiene eventos ni procedimientos propios asociados, ya que su función se limita al almacenamiento estructurado de datos. El control de acceso, validación y procesamiento de la información contenida en esta hoja se gestiona exclusivamente mediante módulos estándar y formularios del sistema.

3.2 Hoja 4

La Hoja 4 forma parte de la estructura interna del programa y se emplea como hoja auxiliar para la gestión de información o procesos de apoyo al sistema. Su utilización está controlada por la lógica del software, sin interacción directa libre por parte del usuario.

3.2.1 Eventos y procedimientos asociados

La Hoja 4 no cuenta con eventos ni procedimientos propios asociados. Las operaciones relacionadas con esta hoja se encuentran controladas desde módulos estándar del proyecto, manteniendo la lógica centralizada y organizada.

3.3 Hoja 5

La Hoja 5 se utiliza como componente interno del sistema para el almacenamiento o soporte de datos requeridos por los procesos del software. Su presencia responde a necesidades internas del programa y no a interacción directa del usuario final.

3.3.1 Eventos y procedimientos asociados

La Hoja 5 no contiene eventos ni procedimientos propios asociados. Su función se limita al soporte interno del sistema, mientras que la lógica de procesamiento y control se ejecuta mediante código VBA ubicado en módulos estándar y formularios.

4. Código de UserForms

Los formularios de usuario (UserForms) constituyen la interfaz principal del programa de cómputo y permiten controlar la interacción del usuario con el sistema. Mediante estos formularios se gestiona la navegación, la ejecución de procesos y la presentación de información, sustituyendo la interacción directa con las hojas de cálculo y reforzando el comportamiento del archivo como una aplicación de software autónoma desarrollada en VBA.

4.1 UserForm frmInicio

El formulario frmInicio funciona como la pantalla inicial del sistema y representa el punto de entrada del usuario al programa. Desde este formulario se controla el acceso a las funcionalidades principales, la ejecución de procesos y la visualización de información, centralizando la interacción del usuario dentro de un entorno gráfico controlado por programación.

4.1.1 Inicialización del formulario

La inicialización del formulario se ejecuta al momento de cargarse el UserForm y permite establecer el estado inicial de la interfaz, configurar controles y preparar el entorno necesario para la interacción del usuario. Este proceso asegura que el formulario se presente de manera consistente y controlada desde el inicio de la ejecución.

```
Private Sub UserForm_Initialize()
```

```
On Error GoTo ErrInit
```

```
Me.MultiPage1.Value = 0
```

```
With Me.cmbMetodo
```

```
    .Clear
```

```
    .AddItem "Método de Diferencias Divididas de Newton"
```

```
    .AddItem "Método Directo de Lagrange"
```

```
    .ListIndex = 0
```

```
End With
```

```
' Forzar desconexión de orígenes externos antes de estructurar
```

```
Me.ListBoxDatos.RowSource = ""
```

```
Call InicializarEstructuraLista
```

```
On Error Resume Next
```

```
Me.Caption = modConfiguracion.ObtenerFirmaSoftware()
```

```
If Err.Number <> 0 Then Me.Caption = TITULO_SISTEMA
```

```
On Error GoTo 0
```

```
Exit Sub
```

```
ErrInit:
```

```
MsgBox "Error crítico al inicializar la interfaz: " & Err.Description, vbCritical,  
TITULO_SISTEMA
```

```
End Sub
```

4.1.2 Eventos de controles

```
Private Sub btnGenerarPDF_Click()
```

```
Call GenerarReportePDF(Me)
```

```
End Sub
```

```
Private Sub btnEstructuralInterna_Click()
```

```
Call modSistema.MostrarEstructuralInterna
```

```
DoEvents
```

```
MsgBox ObtenerEstructuralInternaSistema(), vbInformation, "Estructura Interna del  
Sistema"
```

```
End Sub
```

```
Private Sub btnVerEstructura_Click()
```

```
On Error GoTo ErrorHandler
```

```
Application.Visible = True
```

```
Application.WindowState = xlNormal
```

```
Application.ScreenUpdating = True
```

```
MsgBox ObtenerEstructuralInternaSistema(), vbInformation, "Estructura Interna del Sistema"
```

```
DoEvents
```

```
Application.VBE.MainWindow.Visible = True
```

```
Exit Sub
```

```
ErrorHandler:
```

```
MsgBox "Error: " & Err.Description, vbCritical
```

```
End Sub
```

```
Private Sub cmdAgregarDatos_Click()
```

```
Call AgregarRegistroMuestreo
```

```
End Sub
```

```
Private Sub cmdBorrarUltimo_Click()
```

```
Call RemoverUltimoRegistro
```

```
End Sub
```

```
Private Sub btnCalcular_Click()  
    Call ProcesarCalcularInterpolacion  
    Me.MultiPage1.Value = 1  
End Sub
```

```
Private Sub btnAnalisisEstadistico_Click()  
    ' (TODO el contenido completo del procedimiento,  
    ' sin recortar ni resumir)  
End Sub
```

```
Private Sub btnLimpiarTotal_Click()  
    ' (procedimiento completo)  
End Sub
```

```
Private Sub btnSalir_Click()  
    ' (procedimiento completo)  
End Sub
```

```
Private Sub MultiPage1_Change()  
End Sub
```

4.1.3 Procedimientos internos

Los métodos auxiliares que NO son eventos

```
Private Sub InicializarEstructuraLista()
```

```
    With Me.ListBoxDatos
```

```
        .Clear
```

```
        .ColumnCount = 4
```

```
        .ColumnWidths = "80 pt;90 pt;80 pt;80 pt"
```

```
        .ColumnHeads = False
```

```
        .AddItem
```

```
        .List(0, 0) = "Tramo (Km)"
```

```
        .List(0, 1) = "Velocidad (m/s)"
```

```
        .List(0, 2) = "Tiempo (h)"
```

```
        .List(0, 3) = "SST (mg/L)"
```

```
    End With
```

```
End Sub
```

```
Public Sub AgregarRegistroMuestreo()
```

```
    ' (procedimiento completo)
```

```
End Sub
```

```
Public Sub RemoverUltimoRegistro()
```

```
    ' (procedimiento completo)
```

End Sub

Public Sub ProcesarCalcularInterpolacion()

' (procedimiento completo)

End Sub

Private Function ObtenerEstructuralInternaSistema() As String

' (función completa)

End Function

5. Código de módulos estándar

Los módulos estándar concentran la lógica central del programa de cómputo, incluyendo algoritmos matemáticos, validaciones, control de procesos y generación de resultados. Esta estructura permite separar claramente la interfaz de usuario de los procesos internos, reforzando la naturaleza del sistema como software desarrollado mediante VBA.

5.1 modAlgoritmos

El módulo modAlgoritmos contiene rutinas algorítmicas auxiliares utilizadas por el sistema para apoyar procesos de cálculo y análisis numérico. Estas rutinas encapsulan procedimientos reutilizables que contribuyen al funcionamiento matemático del software.

Option Explicit

```
'  
=====
```

' MÓDULO: modAlgoritmos

' CAPA: Núcleo de Computación Numérica y Resolución Polinomial

' PROVEE: Algoritmo puro de Interpolación por el Método Directo de Lagrange.

```
'  
=====
```

''' <summary>

''' Ejecuta la interpolación polinomial mediante el método clásico de Lagrange.

''' Desarrollado con control activo de desbordamiento aritmético de punto flotante.

''' </summary>

''' <param name="VectorX">Arreglo unidimensional verificado de abscisas conocidas [Tramo (Km)].</param>

''' <param name="VectorY">Arreglo unidimensional verificado de ordenadas conocidas [SST o Velocidad].</param>

''' <param name="XTarget">El valor numérico de la variable independiente a evaluar (Km).</param>

''' <param name="OutResultado">Variable por referencia que almacenará el valor interpolado final.</param>

''' <returns>True si el polinomio se construyó y evaluó de forma exitosa; False ante fallas matemáticas.</returns>

Public Function CalcularLagrange(ByRef VectorX() As Double, ByRef VectorY() As Double, ByVal XTarget As Double, ByRef OutResultado As Double) As Boolean

Dim n As Long

Dim i As Long, j As Long

Dim L_j As Double

Dim acumuladorPolinomial As Double

Dim lb As Long, ub As Long

' Inicialización preventiva del resultado por seguridad

OutResultado = 0#

On Error GoTo ErrorMatematicoLagrange

' Captura de límites de los vectores dinámicos

lb = LBound(VectorX)

ub = UBound(VectorX)

n = ub - lb + 1

' Validación de frontera de seguridad analítica para control de ejecución

If n < 2 Then GoTo ErrorMatematicoLagrange

acumuladorPolinomial = 0#

```

' Lazo externo: Sumatoria de los términos  $L_i(x) * y_i$ 
For i = lb To ub

     $L_i = 1$  # ' Inicialización del producto productorio para el nodo actual

    ' Lazo interno: Productorio de los coeficientes de Lagrange
    For j = lb To ub

        If i <> j Then

            ' Protección secundaria explícita contra divisiones indeterminadas o idénticas
            If Abs(VectorX(i) - VectorX(j)) < modConfiguracion.LIMIT_EPSILON Then
                GoTo ErrorMatematicoLagrange
            End If

            ' Cálculo acumulativo del término  $L_i$  (Productorio)
             $L_i = L_i * (XTarget - VectorX(j)) / (VectorX(i) - VectorX(j))$ 

        End If

    Next j

    ' Suma del término ponderado actual al polinomio global
    acumuladorPolinomial = acumuladorPolinomial + ( $L_i * VectorY(i)$ )

Next i

' Asignación segura del resultado si la ejecución fue limpia
OutResultado = acumuladorPolinomial

CalcularLagrange = True

Exit Function

```

```
ErrorMatematicoLagrange:
```

```
' En caso de un desbordamiento numérico o error imprevisto de punto flotante
```

```
OutResultado = 0#
```

```
CalcularLagrange = False
```

```
End Function
```

5.2 modAppCore

El módulo modAppCore funciona como núcleo central de procesamiento del sistema, coordinando la ejecución de algoritmos, la validación de datos y el flujo general de cálculo. Este módulo actúa como intermediario entre la interfaz y los motores matemáticos del programa.

```
Option Explicit
```

```
'
```

```
=====
```

```
' MÓDULO: modAppCore
```

```
' CAPA: Núcleo de Coordinación y Control Operativo (Orquestador)
```

```
' PROVEE: El puente de ejecución centralizado entre la interfaz y los algoritmos.
```

```
'
```

```
=====
```

```
''' <summary>
```

```
''' Orquesta el flujo de ejecución completo para calcular la interpolación.
```

```
''' </summary>
```

```

''' <param name="VectorX">Arreglo con los datos de Tramo (Km).</param>

''' <param name="VectorY">Arreglo con los datos de SST (mg/L) o Velocidad
(m/s).</param>

''' <param name="XTarget">Ubicación kilométrica a evaluar.</param>

''' <param name="UsarNewton">True si se seleccionó Newton; False si se seleccionó
Lagrange.</param>

''' <param name="OutResultado">Retorna el valor numérico interpolado.</param>

''' <param name="OutGrado">Retorna el grado del polinomio calculado.</param>

''' <param name="OutError">Retorna el error aproximado estimado.</param>

''' <param name="OutMensajeError">Retorna la descripción detallada en caso de
falla.</param>

''' <returns>True si todo el proceso se ejecutó con verosimilitud científica; False si
falló.</returns>

```

```

Public Function EjecutarProcesamientoCentral(ByRef VectorX() As Double, _

```

```

    ByRef VectorY() As Double, _

```

```

    ByVal XTarget As Double, _

```

```

    ByVal usarNewton As Boolean, _

```

```

    ByRef OutResultado As Double, _

```

```

    ByRef OutGrado As Long, _

```

```

    ByRef OutError As Double, _

```

```

    ByRef OutMensajeError As String) As Boolean

```

```

    Dim datosValidos As Boolean

```

```

    Dim exitoCalculo As Boolean

```

```

' Inicialización por seguridad

```

```

OutResultado = 0#

```

```

OutGrado = 0

```

```

OutError = 0#

OutMensajeError = ""

On Error GoTo FalloCriticoCore

' 1. Control de Paso por la Capa de Validaciones

datosValidos = modValidaciones.ValidarMatricesEntrada(VectorX, VectorY,
OutMensajeError)

If Not datosValidos Then

    ' Se detiene el proceso si no pasa los filtros sanitarios de datos

    EjecutarProcesamientoCentral = False

    Exit Function

End If

' 2. Desvío del flujo según el algoritmo seleccionado por el usuario

If usarNewton Then

    ' Ejecuta el motor de Diferencias Divididas de Newton

    exitoCalculo = modInterpolacionNewton.CalcularNewtonPro(VectorX, VectorY,
XTarget, OutResultado, OutGrado, OutError)

    If Not exitoCalculo Then

        OutMensajeError = "Fallo matemático en la matriz piramidal de Newton."

        EjecutarProcesamientoCentral = False

        Exit Function

    End If

Else

    ' Ejecuta el motor directo de Lagrange

```

```
    exitoCalculo = modAlgoritmos.CalcularLagrange(VectorX, VectorY, XTarget, OutResultado)
```

```
    If Not exitoCalculo Then
```

```
        OutMensajeError = "Fallo numérico en el producto productorio de Lagrange."
```

```
        EjecutarProcesamientoCentral = False
```

```
        Exit Function
```

```
    End If
```

```
    ' Lagrange puro no genera tabla de diferencias divididas, se asignan métricas base
```

```
    OutGrado = UBound(VectorX) - LBound(VectorX)
```

```
    OutError = 0#
```

```
End If
```

```
    ' Operación exitosa
```

```
    OutMensajeError = "Cálculo integrado correctamente."
```

```
    EjecutarProcesamientoCentral = True
```

```
    Exit Function
```

```
FalloCriticoCore:
```

```
    OutMensajeError = "Fallo imprevisto en el núcleo del orquestador: " & Err.Description
```

```
    EjecutarProcesamientoCentral = False
```

```
End Function
```

```
''' <summary>
```

```
''' Inicializa la aplicación ocultando el entorno de Excel para control absoluto de la UI.
```

```
''' </summary>
```

```
Public Sub ArrancarAplicacion()
```

```

' Ocultar el entorno de Excel para simular software independiente
Application.Visible = False

' Invocar de forma activa y limpia el formulario de la interfaz
frmInicio.Show

End Sub

''' <summary>
''' Restaura el entorno de Excel de forma segura para auditorías de INDAUTOR.
''' </summary>

Public Sub MostrarEntornoExcel()

    Application.Visible = True

End Sub

```

5.3 modCapturaDatos

El módulo modCapturaDatos se encuentra definido como parte de la arquitectura del sistema con la finalidad de centralizar, en futuras versiones, rutinas relacionadas con la gestión especializada de captura de datos. En la versión actual del programa, las operaciones de captura se encuentran integradas directamente en los procedimientos del formulario de interfaz, por lo que este módulo no contiene código ejecutable.

5.4 modConfiguracion

El módulo modConfiguracion concentra parámetros de configuración del sistema, así como funciones relacionadas con identificación del software, ajustes internos y control de elementos generales necesarios para la correcta operación del programa.

Option Explicit

```
'  
=====
```

' SOFTWARE: InterpolacionNumericaPro1

' TIPO DE OBRA: Programa de Computación (Software)

' TITULAR DE LOS DERECHOS: [Tu Nombre Completo o Razón Social]

' PAÍS DE ORIGEN: México

```
'  
' CONTROL DE VERSIONES:  
' Versión 1.0.0 (Lanzamiento Inicial) - Estructura de Arquitectura Modular  
'  
=====
```

' --- CONSTANTES DE IDENTIFICACIÓN ---

Public Const NOMBRE_APP As String = "InterpolacionNumericaPro1"

Public Const VERSION_APP As String = "1.0.0"

Public Const DISTRIBUCION As String = "Licencia Pro / Registro INDAUTOR"

Public Const AUTOR_SOFTWARE As String = "[Tu Nombre Completo]"

Public Const REGISTRO_ANIO As String = "2026"

' --- CONSTANTES MATEMÁTICAS GLOBALES ---

' Evita errores de desbordamiento o divisiones matemáticas indefinidas

Public Const LIMIT_EPSILON As Double = 0.000000000001

```
Public Const CAPACIDAD_MAX_MATRIZ As Long = 1000
```

```
' --- MAPEO DE INDICES PARA LA LISTA DE DATOS (ListBox) ---
```

```
' Modela la estructura exacta de la interfaz gráfica del usuario
```

```
Public Const COL_TRAMO As Long = 0 ' Columna 1: Tramo (Km)
```

```
Public Const COL_VELOCIDAD As Long = 1 ' Columna 2: Velocidad (m/s)
```

```
Public Const COL_TIEMPO As Long = 2 ' Columna 3: Tiempo (h)
```

```
Public Const COL_SST As Long = 3 ' Columna 4: SST (mg/L)
```

```
,
```

```
=====
```

```
' FUNCIONES PÚBLICAS DE CONFIGURACIÓN
```

```
,
```

```
=====
```

```
''' <summary>
```

```
''' Devuelve el bloque de firma digital de autoría para los cuadros de diálogo.
```

```
''' </summary>
```

```
Public Function ObtenerFirmaSoftware() As String
```

```
    Dim firma As String
```

```
    firma = NOMBRE_APP & " v" & VERSION_APP & vbCrLf & _
```

```
        "Derechos Reservados © " & REGISTRO_ANIO & " - " & AUTOR_SOFTWARE & _  
vbCrLf & _
```

```
        "Desarrollado bajo Normas de Protección de Software Intelectual." & vbCrLf & _
```

```
        "Estado: Fase de Compilación Código Fuente."
```

```
    ObtenerFirmaSoftware = firma
```

```
End Function
```

```
''' <summary>
```

```
''' Verifica si un valor numérico se encuentra dentro del rango de tolerancia cero.
```

```
''' </summary>
```

```
Public Function EsCercanoACero(ByVal Valor As Double) As Boolean
```

```
    If Abs(Valor) < LIMIT_EPSILON Then
```

```
        EsCercanoACero = True
```

```
    Else
```

```
        EsCercanoACero = False
```

```
    End If
```

```
End Function
```

5.5 modEstadistica

El módulo modEstadistica implementa funciones y procedimientos estadísticos utilizados para el análisis de los datos de muestreo, incluyendo cálculos de dispersión, bootstrapping, variabilidad y métricas de diagnóstico empleadas por el sistema.

```
Option Explicit
```

```
,
```

```
=====
```

```
' MÓDULO: modEstadistica
```

```
' CAPA: Análisis Científico de Datos y Control de Certidumbre
```

' PROVEE: Métricas de variabilidad, promedio y dispersión del muestreo.

,

=====

''' <summary>

''' Calcula el promedio aritmético de un vector de datos para establecer líneas base de comparación.

''' </summary>

Public Function CalcularPromedio(ByRef Vector() As Double) As Double

Dim i As Long, lb As Long, ub As Long

Dim suma As Double

lb = LBound(Vector)

ub = UBound(Vector)

suma = 0#

For i = lb To ub

 suma = suma + Vector(i)

Next i

CalcularPromedio = suma / (ub - lb + 1)

End Function

''' <summary>

''' Analiza la dispersión del muestreo mediante la Desviación Estándar Muestral.

''' Se eliminó el acento del nombre para garantizar la compatibilidad de compilación ANSI en VBA.

''' </summary>

Public Function CalcularDesviacionEstandar(ByRef Vector() As Double) As Double

Dim i As Long, lb As Long, ub As Long

Dim promedio As Double, sumaCuadrados As Double, n As Long

lb = LBound(Vector)

ub = UBound(Vector)

n = ub - lb + 1

If n < 2 Then

CalcularDesviacionEstandar = 0#

Exit Function

End If

promedio = CalcularPromedio(Vector)

sumaCuadrados = 0#

For i = lb To ub

sumaCuadrados = sumaCuadrados + ((Vector(i) - promedio) ^ 2)

Next i

' Desviación estándar muestral (n - 1) para máxima verosimilitud estadística

CalcularDesviacionEstandar = Sqr(sumaCuadrados / (n - 1))

End Function

```

''' <summary>

''' Genera un diagnóstico completo de confianza para el botón de Análisis Estadístico.

''' </summary>

''' <param name="VectorX">Arreglo de tramos (Km).</param>

''' <param name="VectorY">Arreglo de la variable de estudio (SST o
Velocidad).</param>

''' <param name="OutVarianza">Retorna la varianza de los datos analizados.</param>

''' <param name="OutRangoEfectivo">Retorna la distancia total cubierta por el
muestreo (Km).</param>

Public Sub AuditarMuestreoPro(ByRef VectorX() As Double, _
    ByRef VectorY() As Double, _
    ByRef OutVarianza As Double, _
    ByRef OutRangoEfectivo As Double)

    Dim lb As Long, ub As Long

    Dim desvEst As Double

    Dim minX As Double, maxX As Double

    Dim i As Long

    lb = LBound(VectorX)
    ub = UBound(VectorX)

    ' 1. Cálculo de varianza basada en la desviación estándar corregida sin acentos
    desvEst = CalcularDesviacionEstandar(VectorY)
    OutVarianza = desvEst ^ 2

    ' 2. Cálculo del rango efectivo (Amplitud geográfica del tramo analizado)

```

```

minX = VectorX(lb)
maxX = VectorX(lb)
For i = lb To ub
    If VectorX(i) < minX Then minX = VectorX(i)
    If VectorX(i) > maxX Then maxX = VectorX(i)
Next i

OutRangoEfectivo = maxX - minX
End Sub

Public Function CalcularCoefVariacion(ByRef Vector() As Double) As Double
    Dim prom As Double
    Dim desv As Double

    prom = CalcularPromedio(Vector)
    desv = CalcularDesviacionEstandar(Vector)

    If prom <> 0 Then
        CalcularCoefVariacion = desv / prom
    Else
        CalcularCoefVariacion = 0
    End If
End Function

Public Function CalcularDensidadMuestreo(ByRef VectorX() As Double) As Double
    Dim n As Long

```

```

Dim rango As Double

n = UBound(VectorX) - LBound(VectorX) + 1
rango = VectorX(UBound(VectorX)) - VectorX(LBound(VectorX))

If rango <> 0 Then
    CalcularDensidadMuestreo = n / rango
Else
    CalcularDensidadMuestreo = 0
End If
End Function

Public Function ValidacionCruzadaError(ByRef X() As Double, ByRef Y() As Double) As Double

    Dim i As Long, j As Long
    Dim n As Long
    Dim errorTotal As Double
    Dim xTemp() As Double, yTemp() As Double
    Dim yEstimado As Double

    n = UBound(X) - LBound(X) + 1
    errorTotal = 0#

    For i = LBound(X) To UBound(X)

```

```
ReDim xTemp(0 To n - 2)
```

```
ReDim yTemp(0 To n - 2)
```

```
Dim k As Long
```

```
k = 0
```

```
For j = LBound(X) To UBound(X)
```

```
    If j <> i Then
```

```
        xTemp(k) = X(j)
```

```
        yTemp(k) = Y(j)
```

```
        k = k + 1
```

```
    End If
```

```
Next j
```

```
' Interpolación con Lagrange
```

```
Call modAlgoritmos.CalcularLagrange(xTemp, yTemp, X(i), yEstimado)
```

```
errorTotal = errorTotal + Abs(yEstimado - Y(i))
```

```
Next i
```

```
ValidacionCruzadaError = errorTotal / n
```

```
End Function
```

```
Public Sub BootstrapIntervalo(ByRef Y() As Double, _
```

```
ByRef lower As Double, _  
ByRef upper As Double)
```

```
Dim i As Long, j As Long
```

```
Dim n As Long
```

```
Dim muestras() As Double
```

```
Dim resultados() As Double
```

```
Dim prom As Double
```

```
Randomize
```

```
n = UBound(Y) - LBound(Y) + 1
```

```
ReDim resultados(1 To 100) ' 100 simulaciones
```

```
For i = 1 To 100
```

```
    prom = 0#
```

```
    For j = 1 To n
```

```
        prom = prom + Y(Int(Rnd() * n) + 1)
```

```
    Next j
```

```
    resultados(i) = prom / n
```

```
Next i
```

```

' Ordenar
Call QuickSort(resultados, LBound(resultados), UBound(resultados))

lower = resultados(5) ' aprox 5%
upper = resultados(95) ' aprox 95%

End Sub

Public Sub QuickSort(arr() As Double, ByVal first As Long, ByVal last As Long)

    Dim i As Long, j As Long
    Dim pivot As Double, temp As Double

    i = first
    j = last
    pivot = arr((first + last) \ 2)

    Do While i <= j
        Do While arr(i) < pivot
            i = i + 1
        Loop
        Do While arr(j) > pivot
            j = j - 1
        Loop
        If i <= j Then

```

```

        temp = arr(i)
        arr(i) = arr(j)
        arr(j) = temp
        i = i + 1
        j = j - 1
    End If
Loop

If first < j Then Call QuickSort(arr, first, j)
If i < last Then Call QuickSort(arr, i, last)

End Sub

```

5.6 modInterpolacionNewton

El módulo modInterpolacionNewton contiene la implementación del algoritmo de interpolación polinómica de Newton, el cual constituye el fundamento matemático principal del programa. Este módulo representa el núcleo algorítmico original del sistema.

Option Explicit

```

'
=====
=====

' MÓDULO: modInterpolacionNewton
' CAPA: Núcleo de Computación Numérica Avanzada
' PROVEE: Algoritmo de Diferencias Divididas de Newton y evaluación de error.

```

```

'
=====
=====

''' <summary>

''' Ejecuta la interpolación polinomial mediante diferencias divididas de Newton.

''' Además de evaluar el punto, calcula métricas para el Análisis Estadístico.

''' </summary>

''' <param name="VectorX">Arreglo de abscisas [Tramo (Km)].</param>

''' <param name="VectorY">Arreglo de ordenadas [SST (mg/L) o Velocidad
(m/s)].</param>

''' <param name="XTarget">La ubicación kilométrica a evaluar (Km).</param>

''' <param name="OutResultado">Variable de salida para el valor interpolado
aproximado.</param>

''' <param name="OutGradoPolinomio">Variable de salida que indica el grado del
polinomio generado (n-1).</param>

''' <param name="OutErrorAproximado">Variable de salida que estima la estabilidad
del cálculo basada en el último término.</param>

''' <returns>True si el cálculo es exitoso y verosímil; False ante anomalías
numéricas.</returns>

Public Function CalcularNewtonPro(ByRef VectorX() As Double, _
    ByRef VectorY() As Double, _
    ByVal XTarget As Double, _
    ByRef OutResultado As Double, _
    ByRef OutGradoPolinomio As Long, _
    ByRef OutErrorAproximado As Double) As Boolean

    Dim n As Long

    Dim i As Long, j As Long

```

Dim lb As Long, ub As Long

Dim tablaDD() As Double

Dim factorTermino As Double

Dim acumuladorPolinomial As Double

' Inicialización de variables de salida por seguridad

OutResultado = 0#

OutGradoPolinomio = 0

OutErrorAproximado = 0#

On Error GoTo FalloMatematicoNewton

lb = LBound(VectorX)

ub = UBound(VectorX)

n = ub - lb + 1

' Definición del Grado del Polinomio (Propiedad fundamental: $n - 1$)

OutGradoPolinomio = n - 1

' Dimensionar matriz piramidal para almacenar las diferencias divididas

ReDim tablaDD(1 To n, 1 To n)

' 1. Carga de la primera columna con los valores experimentales de Y

For i = 1 To n

 tablaDD(i, 1) = VectorY(lb + i - 1)

Next i

```

' 2. Algoritmo iterativo: Construcción de la tabla de diferencias divididas

For j = 2 To n

    For i = 1 To n - j + 1

        ' Protección explícita contra divisiones por cero en nodos adyacentes

        If Abs(VectorX(lb + i + j - 2) - VectorX(lb + i - 1)) <
modConfiguracion.LIMIT_EPSILON Then

            GoTo FalloMatematicoNewton

        End If

        ' Ecuación fundamental de Newton:

        tablaDD(i, j) = (tablaDD(i + 1, j - 1) - tablaDD(i, j - 1)) / (VectorX(lb + i + j - 2) -
VectorX(lb + i - 1))

        Next i

    Next j

' 3. Evaluación del Polinomio de Newton en el punto XTarget

acumuladorPolinomial = tablaDD(1, 1)

factorTermino = 1#

For i = 1 To n - 1

    ' Multiplicación sucesiva de binomios: (x - x_0)(x - x_1)...(x - x_{n-1})

    factorTermino = factorTermino * (XTarget - VectorX(lb + i - 1))

    ' Sumatoria del término ponderado por su coeficiente de diferencias divididas

    acumuladorPolinomial = acumuladorPolinomial + (tablaDD(1, i + 1) *
factorTermino)

```

Next i

' 4. Estimación del Error Aproximado (Verosimilitud Científica)

' Se calcula utilizando el último monomio evaluable para analizar la estabilidad local

If n >= 2 Then

 OutErrorAproximado = Abs(tablaDD(1, n) * factorTermino)

Else

 OutErrorAproximado = 0#

End If

' Asignación final de resultados validados

OutResultado = acumuladorPolinomial

CalcularNewtonPro = True

Exit Function

FalloMatematicoNewton:

OutResultado = 0#

OutGradoPolinomio = 0

OutErrorAproximado = -1# ' Bandera de error matemático

CalcularNewtonPro = False

End Function

5.7 modReportes

El módulo modReportes se encarga de la generación de salidas del sistema, tales como reportes estructurados y documentos derivados de los resultados del

procesamiento, permitiendo la presentación formal de la información generada por el software.

Option Explicit

```
'  
=====
```

' MÓDULO: modInterpolacionNewton

' CAPA: Núcleo de Computación Numérica Avanzada

' PROVEE: Algoritmo de Diferencias Divididas de Newton y evaluación de error.

```
'  
=====
```

''' <summary>

''' Ejecuta la interpolación polinomial mediante diferencias divididas de Newton.

''' Además de evaluar el punto, calcula métricas para el Análisis Estadístico.

''' </summary>

''' <param name="VectorX">Arreglo de abscisas [Tramo (Km)].</param>

''' <param name="VectorY">Arreglo de ordenadas [SST (mg/L) o Velocidad (m/s)].</param>

''' <param name="XTarget">La ubicación kilométrica a evaluar (Km).</param>

''' <param name="OutResultado">Variable de salida para el valor interpolado aproximado.</param>

''' <param name="OutGradoPolinomio">Variable de salida que indica el grado del polinomio generado (n-1).</param>

''' <param name="OutErrorAproximado">Variable de salida que estima la estabilidad del cálculo basada en el último término.</param>

''' <returns>True si el cálculo es exitoso y verosímil; False ante anomalías numéricas.</returns>

Public Function CalcularNewtonPro(ByRef VectorX() As Double, _

ByRef VectorY() As Double, _

ByVal XTarget As Double, _

ByRef OutResultado As Double, _

ByRef OutGradoPolinomio As Long, _

ByRef OutErrorAproximado As Double) As Boolean

Dim n As Long

Dim i As Long, j As Long

Dim lb As Long, ub As Long

Dim tablaDD() As Double

Dim factorTermino As Double

Dim acumuladorPolinomial As Double

' Inicialización de variables de salida por seguridad

OutResultado = 0#

OutGradoPolinomio = 0

OutErrorAproximado = 0#

On Error GoTo FalloMatematicoNewton

lb = LBound(VectorX)

ub = UBound(VectorX)

n = ub - lb + 1

```

' Definición del Grado del Polinomio (Propiedad fundamental: n - 1)
OutGradoPolinomio = n - 1

' Dimensionar matriz piramidal para almacenar las diferencias divididas
ReDim tablaDD(1 To n, 1 To n)

' 1. Carga de la primera columna con los valores experimentales de Y
For i = 1 To n
    tablaDD(i, 1) = VectorY(lb + i - 1)
Next i

' 2. Algoritmo iterativo: Construcción de la tabla de diferencias divididas
For j = 2 To n
    For i = 1 To n - j + 1
        ' Protección explícita contra divisiones por cero en nodos adyacentes
        If Abs(VectorX(lb + i + j - 2) - VectorX(lb + i - 1)) <
modConfiguracion.LIMIT_EPSILON Then
            GoTo FalloMatematicoNewton
        End If

        ' Ecuación fundamental de Newton:
        tablaDD(i, j) = (tablaDD(i + 1, j - 1) - tablaDD(i, j - 1)) / (VectorX(lb + i + j - 2) -
VectorX(lb + i - 1))
    Next i
Next j

' 3. Evaluación del Polinomio de Newton en el punto XTarget

```

```

acumuladorPolinomial = tablaDD(1, 1)

factorTermino = 1#

For i = 1 To n - 1
    ' Multiplicación sucesiva de binomios: (x - x_0)(x - x_1)...(x - x_{n-1})
    factorTermino = factorTermino * (XTarget - VectorX(lb + i - 1))

    ' Sumatoria del término ponderado por su coeficiente de diferencias divididas
    acumuladorPolinomial = acumuladorPolinomial + (tablaDD(1, i + 1) *
factorTermino)
Next i

' 4. Estimación del Error Aproximado (Verosimilitud Científica)
' Se calcula utilizando el último monomio evaluable para analizar la estabilidad local
If n >= 2 Then
    OutErrorAproximado = Abs(tablaDD(1, n) * factorTermino)
Else
    OutErrorAproximado = 0#
End If

' Asignación final de resultados validados
OutResultado = acumuladorPolinomial
CalcularNewtonPro = True
Exit Function

```

FalloMatematicoNewton:

```

OutResultado = 0#
OutGradoPolinomio = 0
OutErrorAproximado = -1# ' Bandera de error matemático
CalcularNewtonPro = False
End Function

```

5.8 modSemiotica

El módulo modSemiotica implementa la lógica de interpretación visual y semántica de los resultados, asignando criterios de color, diagnóstico y representación simbólica para apoyar la comprensión de los resultados por parte del usuario.

Option Explicit

```

'
=====
=====
' MÓDULO: modSemiotica
' CAPA: Interpretación Lingüística y Semiótica Visual
' PROVEE: Gestión de colores institucionales y semántica de alertas para Page 2.
'
=====
=====

' --- PALETA DE COLORES SEMIÓTICOS (Constantes de Color de Fuente de Excel/VBA)
---
```

Public Const COLOR_ALTA_CONFIANZA As Long = 32768 ' Verde Oscuro (Puntos Internos Estables)

Public Const COLOR_ADVERTENCIA As Long = 32896 ' Naranja/Amarillo Viejo (Extrapolaciones)

Public Const COLOR_CRITICO As Long = 255 ' Rojo Fuerte (Valores Fuera de Límite o Errores)

''' <summary>

''' Devuelve el código de color numérico (Long) adaptado a la verosimilitud de la ubicación evaluada.

''' </summary>

''' <param name="VectorX">Arreglo de tramos conocidos.</param>

''' <param name="XTarget">Ubicación kilométrica a evaluar.</param>

''' <returns>Código numérico de color para aplicar a la fuente de los controles.</returns>

Public Function DefinirColorPorMuestreo(ByRef VectorX() As Double, ByVal XTarget As Double) As Long

 ' Evaluar si el punto requiere una extrapolación o es interno

 If modValidaciones.EsPuntoInterno(VectorX, XTarget) Then

 DefinirColorPorMuestreo = COLOR_ALTA_CONFIANZA

 Else

 DefinirColorPorMuestreo = COLOR_ADVERTENCIA

 End If

End Function

''' <summary>

''' Genera un veredicto o interpretación textual automática sobre el estado del cálculo de SST.

```
"" </summary>
```

```
"" <param name="ValorSST">Resultado numérico interpolado de Sólidos Suspendidos Totales.</param>
```

```
"" <param name="OutColorTexto">Asigna por referencia el color crítico si se violan normas ecológicas.</param>
```

```
"" <returns>Cadena de texto explicativa para la interfaz del usuario.</returns>
```

```
Public Function InterpretarResultadoSST(ByVal ValorSST As Double, ByRef OutColorTexto As Long) As String
```

```
    Dim diagnostico As String
```

```
    ' Suposición de umbral ecológico estándar en ingeniería ambiental: 150 mg/L
```

```
    If ValorSST < 0 Then
```

```
        OutColorTexto = COLOR_CRITICO
```

```
        diagnostico = "ANOMALÍA: El modelo matemático arroja un valor negativo. Revise la dispersión de sus datos de origen."
```

```
    ElseIf ValorSST > 150# Then
```

```
        OutColorTexto = COLOR_CRITICO
```

```
        diagnostico = "ALERTA CRÍTICA: La concentración estimada supera los límites normativos permitidos (>150 mg/L)."
```

```
    Else
```

```
        ' Mantiene el color base determinado por la confianza de ubicación
```

```
        diagnostico = "OPERACIÓN NORMAL: Concentración estimada dentro de los parámetros esperados de estabilidad hidráulica."
```

```
    End If
```

```
    InterpretarResultadoSST = diagnostico
```

```
End Function
```

5.9. modSistema

El módulo modSistema concentra funciones de control general y soporte estructural del programa de cómputo, permitiendo la coordinación entre la interfaz de usuario, los módulos lógicos y la configuración interna del sistema. Este módulo facilita tareas de auditoría, visualización de la arquitectura interna y control del estado global de la aplicación, reforzando la trazabilidad y la transparencia técnica del software.

Option Explicit

```
'
=====
=====

' MÓDULO: modSemiotica

' CAPA: Interpretación Lingüística y Semiótica Visual

' PROVEE: Gestión de colores institucionales y semántica de alertas para Page 2.

'
=====
=====

' --- PALETA DE COLORES SEMIÓTICOS (Constantes de Color de Fuente de Excel/VBA)
---

Public Const COLOR_ALTA_CONFianza As Long = 32768      ' Verde Oscuro (Puntos
Internos Estables)

Public Const COLOR_ADVERTENCIA As Long = 32896         ' Naranja/Amarillo Viejo
(Extrapolaciones)

Public Const COLOR_CRITICO As Long = 255               ' Rojo Fuerte (Valores Fuera de
Límite o Errores)

''' <summary>
```

''' Devuelve el código de color numérico (Long) adaptado a la verosimilitud de la ubicación evaluada.

''' </summary>

''' <param name="VectorX">Arreglo de tramos conocidos.</param>

''' <param name="XTarget">Ubicación kilométrica a evaluar.</param>

''' <returns>Código numérico de color para aplicar a la fuente de los controles.</returns>

Public Function DefinirColorPorMuestreo(ByRef VectorX() As Double, ByVal XTarget As Double) As Long

 ' Evaluar si el punto requiere una extrapolación o es interno

 If modValidaciones.EsPuntoInterno(VectorX, XTarget) Then

 DefinirColorPorMuestreo = COLOR_ALTA_CONFIANZA

 Else

 DefinirColorPorMuestreo = COLOR_ADVERTENCIA

 End If

End Function

''' <summary>

''' Genera un veredicto o interpretación textual automática sobre el estado del cálculo de SST.

''' </summary>

''' <param name="ValorSST">Resultado numérico interpolado de Sólidos Suspendidos Totales.</param>

''' <param name="OutColorTexto">Asigna por referencia el color crítico si se violan normas ecológicas.</param>

''' <returns>Cadena de texto explicativa para la interfaz del usuario.</returns>

Public Function InterpretarResultadoSST(ByVal ValorSST As Double, ByRef OutColorTexto As Long) As String

```

Dim diagnostico As String

' Suposición de umbral ecológico estándar en ingeniería ambiental: 150 mg/L
If ValorSST < 0 Then
    OutColorTexto = COLOR_CRITICO
    diagnostico = "ANOMALÍA: El modelo matemático arroja un valor negativo. Revise la dispersión de sus datos de origen."
ElseIf ValorSST > 150# Then
    OutColorTexto = COLOR_CRITICO
    diagnostico = "ALERTA CRÍTICA: La concentración estimada supera los límites normativos permitidos (>150 mg/L)."
Else
    ' Mantiene el color base determinado por la confianza de ubicación
    diagnostico = "OPERACIÓN NORMAL: Concentración estimada dentro de los parámetros esperados de estabilidad hidráulica."
End If

InterpretarResultadoSST = diagnostico
End Function

```

5.10 modValidaciones

El módulo modValidaciones contiene rutinas de verificación y control de consistencia de datos, destinadas a prevenir errores de entrada, inconsistencias lógicas y condiciones inválidas durante la ejecución del sistema.

Option Explicit

```

'
=====
=====

' MÓDULO: modSemiotica

' CAPA: Interpretación Lingüística y Semiótica Visual

' PROVEE: Gestión de colores institucionales y semántica de alertas para Page 2.
'
=====
=====

' --- PALETA DE COLORES SEMIÓTICOS (Constantes de Color de Fuente de Excel/VBA)
---

Public Const COLOR_ALTA_CONFIANZA As Long = 32768      ' Verde Oscuro (Puntos
Internos Estables)

Public Const COLOR_ADVERTENCIA As Long = 32896        ' Naranja/Amarillo Viejo
(Extrapolaciones)

Public Const COLOR_CRITICO As Long = 255              ' Rojo Fuerte (Valores Fuera de
Límite o Errores)

''' <summary>

''' Devuelve el código de color numérico (Long) adaptado a la verosimilitud de la
ubicación evaluada.

''' </summary>

''' <param name="VectorX">Arreglo de tramos conocidos.</param>

''' <param name="XTarget">Ubicación kilométrica a evaluar.</param>

''' <returns>Código numérico de color para aplicar a la fuente de los
controles.</returns>

```

```
Public Function DefinirColorPorMuestreo(ByRef VectorX() As Double, ByVal XTarget As Double) As Long
```

```
    ' Evaluar si el punto requiere una extrapolación o es interno
```

```
    If modValidaciones.EsPuntoInterno(VectorX, XTarget) Then
```

```
        DefinirColorPorMuestreo = COLOR_ALTA_CONFIANZA
```

```
    Else
```

```
        DefinirColorPorMuestreo = COLOR_ADVERTENCIA
```

```
    End If
```

```
End Function
```

```
''' <summary>
```

```
''' Genera un veredicto o interpretación textual automática sobre el estado del cálculo de SST.
```

```
''' </summary>
```

```
''' <param name="ValorSST">Resultado numérico interpolado de Sólidos Suspendidos Totales.</param>
```

```
''' <param name="OutColorTexto">Asigna por referencia el color crítico si se violan normas ecológicas.</param>
```

```
''' <returns>Cadena de texto explicativa para la interfaz del usuario.</returns>
```

```
Public Function InterpretarResultadoSST(ByVal ValorSST As Double, ByRef OutColorTexto As Long) As String
```

```
    Dim diagnostico As String
```

```
    ' Suposición de umbral ecológico estándar en ingeniería ambiental: 150 mg/L
```

```
    If ValorSST < 0 Then
```

```
        OutColorTexto = COLOR_CRITICO
```

```
        diagnostico = "ANOMALÍA: El modelo matemático arroja un valor negativo. Revise la dispersión de sus datos de origen."
```

```

ElseIf ValorSST > 150# Then

    OutColorTexto = COLOR_CRITICO

    diagnostico = "ALERTA CRÍTICA: La concentración estimada supera los límites
normativos permitidos (>150 mg/L)."

Else

    ' Mantiene el color base determinado por la confianza de ubicación

    diagnostico = "OPERACIÓN NORMAL: Concentración estimada dentro de los
parámetros esperados de estabilidad hidráulica."

End If

InterpretarResultadoSST = diagnostico

End Function

```

6. FUNDAMENTO MATEMÁTICO INTEGRADO

El presente programa de cómputo integra su fundamento matemático directamente dentro de la lógica de programación desarrollada en VBA, evitando la dependencia de fórmulas visibles en hojas de cálculo. Los métodos numéricos y estadísticos se implementan mediante procedimientos y funciones propias, garantizando que el tratamiento matemático forme parte integral del software y no de la estructura de la hoja.

6.1. Interpolación polinómica de Newton

La interpolación polinómica de Newton es un método numérico que permite estimar valores intermedios de una función a partir de un conjunto discreto de datos, utilizando el esquema de diferencias divididas. Este método es especialmente adecuado para conjuntos de datos no uniformemente espaciados, como ocurre en los muestreos hidráulicos y ambientales abordados por el sistema.

6.2. Fundamento matemático coherente con modInterpolacionNewton

El fundamento matemático descrito se encuentra implementado de forma coherente en el módulo modInterpolacionNewton, donde se programan las rutinas necesarias para el cálculo de diferencias divididas, la construcción del polinomio interpolante y la evaluación del resultado en un punto objetivo. La lógica matemática se ejecuta completamente mediante código VBA, asegurando consistencia entre el modelo teórico y su aplicación computacional.

6.3. Enfoque en estimación de SST

El enfoque matemático del sistema se orienta a la estimación de Sólidos Suspendidos Totales (SST) en tramos fluviales, a partir de datos de muestreo obtenidos en ubicaciones discretas. La interpolación permite estimar concentraciones de SST en puntos no muestreados directamente, apoyando el análisis técnico y la toma de decisiones en contextos de gestión y operación de recursos hídricos, en los tramos del río donde trabajan plantas potabilizadoras de agua potable.

7. CÓDIGO FUENTE NUMERADO

El presente apartado contiene el código fuente completo del programa de cómputo **InterpolacionNumericaPro1**, transcrito de manera íntegra y consecutiva, sin omisiones ni modificaciones, con fines de inspección técnica y pericial ante el Instituto Nacional del Derecho de Autor. La numeración permite identificar de forma precisa cada línea del código que conforma el sistema.

7.1. ThisWorkbook

A continuación se presenta el código correspondiente al objeto **ThisWorkbook**, el cual contiene los eventos que controlan el ciclo de vida del sistema, incluyendo la inicialización y el cierre controlado del programa.

```
Private Sub Workbook_Open()

    ' Ocultar Excel completamente
    Application.Visible = False
    Application.ScreenUpdating = False

    ' Mostrar interfaz principal
    frmInicio.Show

End Sub
```

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)
```

```
    ' Restaurar Excel antes de cerrar
```

```
    Application.Visible = True
```

```
    Application.ScreenUpdating = True
```

```
End Sub
```

7.2. Hoja1

El siguiente bloque corresponde al código asociado al objeto **Hoja1** del libro. En la versión actual del sistema, esta hoja no contiene procedimientos ni eventos programados.

7.3. frmInicio

A continuación, se presenta el código fuente completo del formulario **frmInicio**, el cual constituye la interfaz principal del sistema y concentra los eventos de interacción con el usuario, así como procedimientos internos de soporte y control.

```
'
```

```
=====
```

```
=====
```

```
' INTERFAZ GRÁFICA CONTROLADORA: frmInicio (Capa de Presentación)
```

```
' DESCRIPCIÓN: Gobierna eventos de usuario, validaciones en caliente de cajas de
' texto, cálculo dinámico de física hidráulica y semiótica visual.
' Automatización: Cálculo interno del Tiempo (h) sin intervención del usuario.
' REGISTRO DE PROPIEDAD INTELECTUAL: Optimizado para auditoría de INDAUTOR.
'
=====
=====
```

```
' --- CONSTANTES DE CONTROL INTERNO DE DISEÑO ---
```

```
Private Const TITULO_SISTEMA As String = "Interpolación Numérica Pro"
```

```
Private Sub btnGenerarPDF_Click()
```

```
    Call GenerarReportePDF(Me)
```

```
End Sub
```

```
Private Sub btnEstructuralInterna_Click()
```

```
    Call modSistema.MostrarEstructuralInterna
```

```
    DoEvents ' ?? permite renderizar Excel antes del MsgBox
```

```
    MsgBox ObtenerEstructuralInternaSistema(), vbInformation, "Estructura Interna del
Sistema"
```

End Sub

Private Sub btnVerEstructura_Click()

On Error GoTo ErrorHandler

' 1. Mostrar Excel

Application.Visible = True

Application.WindowState = xlNormal

Application.ScreenUpdating = True

' 2. Mostrar información

MsgBox ObtenerEstructuralInternaSistema(), vbInformation, "Estructura Interna del Sistema"

' 3. ABRIR VBA DESPUÉS DE ACEPTAR

DoEvents

Application.VBE.MainWindow.Visible = True

Exit Sub

ErrorHandler:

MsgBox "Error: " & Err.Description, vbCritical

End Sub

,

=====

' 1. EVENTOS GESTORES DE LOS BOTONES DE COMANDO (INTERFAZ VISUAL)

,

=====

Private Sub cmdAgregarDatos_Click()

 Call AgregarRegistroMuestreo

End Sub

Private Sub cmdBorrarUltimo_Click()

 Call RemoverUltimoRegistro

End Sub

Private Sub btnCalcular_Click()

 Call ProcesarCalcularInterpolacion

 ' Cambiar automáticamente a página de resultados

 Me.MultiPage1.Value = 1

End Sub

```

Private Sub btnAnalisisEstadistico_Click()

    Dim X() As Double
    Dim Y() As Double
    Dim i As Long
    Dim n As Long

    n = Me.ListBoxDatos.ListCount - 1

    If n < 3 Then
        MsgBox "Se requieren al menos 3 datos", vbExclamation
        Exit Sub
    End If

    ReDim X(1 To n)
    ReDim Y(1 To n)

    For i = 1 To n
        X(i) = CDBl(Me.ListBoxDatos.List(i, 0))
        Y(i) = CDBl(Me.ListBoxDatos.List(i, 3))
    Next i

    Dim prom As Double, desv As Double
    Dim varianza As Double

```

```

Dim diagnostico As String
Dim rango As Double
Dim cv As Double
Dim densidad As Double
Dim errorCV As Double
Dim li As Double, ls As Double

prom = CalcularPromedio(Y)
desv = CalcularDesviacionEstandar(Y)
varianza = desv ^ 2

Call AuditarMuestreoPro(X, Y, varianza, rango)

cv = CalcularCoefVariacion(Y)
densidad = CalcularDensidadMuestreo(X)
errorCV = ValidacionCruzadaError(X, Y)

Call BootstrapIntervalo(Y, li, ls)

' =====
' DIAGNÓSTICO AUTOMÁTICO
' =====

If cv < 0.1 Then
    diagnostico = "Muy baja variabilidad. Sistema estable."
ElseIf cv < 0.3 Then
    diagnostico = "Variabilidad moderada. Resultados aceptables."

```

Else

 diagnostico = "Alta variabilidad. Alta incertidumbre en el modelo."

End If

If densidad < 1 Then

 diagnostico = diagnostico & vbCrLf & "Baja densidad de muestreo. Se recomienda recolectar más datos."

End If

If errorCV > 5 Then

 diagnostico = diagnostico & vbCrLf & "Error alto en validación. Revisar calidad de interpolación."

End If

' MOSTRAR RESULTADO

Me.txtReporteEstadistico.Value = _

 "Promedio: " & Format(prom, "0.00") & " mg/L" & vbCrLf & _

 "Desviación: " & Format(desv, "0.00") & vbCrLf & _

 "Varianza: " & Format(varianza, "0.00") & vbCrLf & vbCrLf & _

 "Coeficiente de variación: " & Format(cv * 100, "0.00") & " %" & vbCrLf & _

 "Rango: " & Format(rango, "0.00") & " km" & vbCrLf & _

 "Densidad: " & Format(densidad, "0.00") & " pts/km" & vbCrLf & vbCrLf & _

 "Error validación: ±" & Format(errorCV, "0.00") & vbCrLf & _

 "Intervalo Bootstrap: " & Format(li, "0.00") & " - " & Format(ls, "0.00") & vbCrLf & vbCrLf & _

 "Diagnóstico:" & vbCrLf & diagnostico

End Sub

Private Sub btnLimpiarTotal_Click()

Dim respuesta As VbMsgBoxResult

respuesta = MsgBox("¿Está completamente seguro de que desea eliminar todos los registros capturados y reiniciar la sesión?", _

vbQuestion + vbYesNo + vbDefaultButton2, TITULO_SISTEMA)

If respuesta = vbYes Then

Me.txtTramo.Value = ""

Me.txtVelocidad.Value = ""

Me.txtSST.Value = ""

Call InicializarEstructuraLista

Me.txtTramo.SetFocus

MsgBox "El registro total ha sido vaciado de la memoria con éxito.", vbInformation, TITULO_SISTEMA

End If

End Sub

Private Sub btnSalir_Click()

Dim respuesta As VbMsgBoxResult

respuesta = MsgBox("¿Está seguro que desea salir de la aplicación?", _

vbQuestion + vbYesNo + vbDefaultButton2, _

"Confirmar salida")

```
If respuesta = vbYes Then
```

```
Application.Visible = True
```

```
Application.ScreenUpdating = True
```

```
ThisWorkbook.Close SaveChanges:=True
```

```
End If
```

```
End Sub
```

```
Private Sub MultiPage1_Change()
```

```
End Sub
```

```
'  
=====
```

```
' 2. MÉTODOS DE INICIALIZACIÓN Y CONFIGURACIÓN ESTRUCTURAL
```

```
'  
=====
```

```
Private Sub UserForm_Initialize()
```

```

On Error GoTo ErrInit

Me.MultiPage1.Value = 0

With Me.cmbMetodo

    .Clear

    .AddItem "Método de Diferencias Divididas de Newton"

    .AddItem "Método Directo de Lagrange"

    .ListIndex = 0

End With

' Forzar desconexión de orígenes externos antes de estructurar
Me.ListBoxDatos.RowSource = ""

Call InicializarEstructuraLista

On Error Resume Next

Me.Caption = modConfiguracion.ObtenerFirmaSoftware()

If Err.Number <> 0 Then Me.Caption = TITULO_SISTEMA

On Error GoTo 0

Exit Sub

ErrInit:

    MsgBox "Error crítico al inicializar la interfaz: " & Err.Description, vbCritical,
    TITULO_SISTEMA

End Sub

```

Private Sub InicializarEstructuraLista()

With Me.ListBoxDatos

.Clear

.ColumnCount = 4

.ColumnWidths = "80 pt;90 pt;80 pt;80 pt"

.ColumnHeads = False

.AddItem

.List(0, 0) = "Tramo (Km)"

.List(0, 1) = "Velocidad (m/s)"

.List(0, 2) = "Tiempo (h)"

.List(0, 3) = "SST (mg/L)"

End With

End Sub

,

=====
=====

' 3. MOTORES LÓGICOS DE PROCESAMIENTO DE DATOS Y CÁLCULO

,

=====
=====

Public Sub AgregarRegistroMuestreo()

Dim valTramo As Double, valVelocidad As Double, valSST As Double, valTiempo As Double

Dim filaDestino As Long, tramoAnterior As Double, deltaDistanciaMeters As Double

' 1. Validaciones de Interfaz de Usuario

```
If Trim(Me.txtTramo.Value) = "" Or Trim(Me.txtVelocidad.Value) = "" Or Trim(Me.txtSST.Value) = "" Then
```

```
    MsgBox "Campos incompletos: Para ingresar un punto de muestreo debe llenar obligatoriamente los cuadros de Tramo, Velocidad y SST.", vbExclamation, TITULO_SISTEMA
```

```
    Exit Sub
```

```
End If
```

```
If Not IsNumeric(Me.txtTramo.Value) Or Not IsNumeric(Me.txtVelocidad.Value) Or Not IsNumeric(Me.txtSST.Value) Then
```

```
    MsgBox "Error de formato: Todos los parámetros ingresados deben ser estrictamente numéricos.", vbExclamation, TITULO_SISTEMA
```

```
    Exit Sub
```

```
End If
```

```
valTramo = CDBl(Me.txtTramo.Value)
```

```
valVelocidad = CDBl(Me.txtVelocidad.Value)
```

```
valSST = CDBl(Me.txtSST.Value)
```

```
If valTramo < 0# Or valVelocidad <= 0# Or valSST < 0# Then
```

```
    MsgBox "Error de consistencia física: Los valores no pueden ser negativos, y la velocidad debe ser mayor a cero.", vbCritical, TITULO_SISTEMA
```

```
    Exit Sub
```

```
End If
```

```
' Obtener la dimensión real actual del arreglo del ListBox
```

```
filaDestino = Me.ListBoxDatos.ListCount
```

```

' 2. Inserción Segura de Datos en Filas Consecutivas

If filaDestino = 1 Then

    ' Primera muestra física (Fila indexada 1)

    Me.ListBoxDatos.AddItem

    Me.ListBoxDatos.List(1, 0) = Format(valTramo, "0.000")

    Me.ListBoxDatos.List(1, 1) = Format(valVelocidad, "0.00")

    Me.ListBoxDatos.List(1, 2) = "0.0000"

    Me.ListBoxDatos.List(1, 3) = Format(valSST, "0.00")

Else

    ' Muestras subsecuentes con validación topográfica lineal

    tramoAnterior = CDBl(Me.ListBoxDatos.List(filaDestino - 1, 0))

    If valTramo <= tramoAnterior Then

        MsgBox "Inconsistencia de Trayecto: El Tramo (Km) actual debe denotar un
        avance lineal respecto al punto anterior registrado (" & tramoAnterior & " Km).",
        vbExclamation, TITULO_SISTEMA

        Exit Sub

    End If

    ' Cálculo de transitoriedad física (Tiempo)

    deltaDistanciaMeters = (valTramo - tramoAnterior) * 1000#

    valTiempo = deltaDistanciaMeters / (valVelocidad * 3600#)

    ' Añadir fila al final preservando la memoria interna del control

    Me.ListBoxDatos.AddItem

    Me.ListBoxDatos.List(filaDestino, 0) = Format(valTramo, "0.000")

    Me.ListBoxDatos.List(filaDestino, 1) = Format(valVelocidad, "0.00")

    Me.ListBoxDatos.List(filaDestino, 2) = Format(valTiempo, "0.0000")

```

```

        Me.ListBoxDatos.List(filaDestino, 3) = Format(valSST, "0.00")
    End If

' 3. Limpieza de cajas de texto
Me.txtTramo.Value = ""
Me.txtVelocidad.Value = ""
Me.txtSST.Value = ""
Me.txtTramo.SetFocus
End Sub

Public Sub RemoverUltimoRegistro()

    Dim totalActual As Long

    totalActual = Me.ListBoxDatos.ListCount

    ' Validar que exista al menos una fila de datos (además del encabezado)
    If totalActual <= 1 Then

        MsgBox "No hay datos para eliminar.", vbExclamation, TITULO_SISTEMA

        Exit Sub

    End If

    ' Confirmación opcional (recomendado)
    If MsgBox("¿Desea eliminar el último registro?", vbQuestion + vbYesNo) = vbNo Then

        Exit Sub

    End If

```

```

' Eliminar última fila
Me.ListBoxDatos.RemoveItem totalActual - 1

MsgBox "Último registro eliminado correctamente.", vbInformation, TITULO_SISTEMA

End Sub

Public Sub ProcesarCalcularInterpolacion()
    Dim TotalFilas As Long, datosReales As Long, i As Long
    Dim XTarget As Double, usarNewton As Boolean
    Dim arrX() As Double, arrY() As Double
    Dim resCalculado As Double, gradoPolinomio As Long, errorAproximado As Double
    Dim msgSalidaCore As String, colorSemiotico As Long, analisisTexto As String
    Dim reporteTextoFinal As String

    TotalFilas = Me.ListBoxDatos.ListCount
    datosReales = TotalFilas - 1

    ' Validar cantidad mínima de datos
    If datosReales < 2 Then
        MsgBox "Cálculo Interrumpido: Datos insuficientes en la lista." & vbCrLf & _
            "Se requiere un mínimo de 2 líneas de datos reales para construir un polinomio interpolante.", vbExclamation, TITULO_SISTEMA
    End If
    Exit Sub
End If

```

```

' Validar caja de texto del parámetro objetivo

If Trim(Me.txtUbicacionEvaluar.Value) = "" Or Not
IsNumeric(Me.txtUbicacionEvaluar.Value) Then

    MsgBox "Error de configuración: Debe ingresar un valor numérico válido en el
campo de ubicación a evaluar (Km) antes de proceder.", vbExclamation,
TITULO_SISTEMA

    Exit Sub

End If

XTarget = CDbl(Me.txtUbicacionEvaluar.Value)

ReDim arrX(1 To datosReales)
ReDim arrY(1 To datosReales)

For i = 1 To datosReales
    arrX(i) = CDbl(Me.ListBoxDatos.List(i, 0))
    arrY(i) = CDbl(Me.ListBoxDatos.List(i, 3))
Next i

If Me.cmbMetodo.ListIndex = 0 Then
    usarNewton = True
Else
    usarNewton = False
End If

On Error GoTo ErrMathCore

```

' Ejecución del Núcleo Centralizador de Procesamiento

If Not modAppCore.EjecutarProcesamientoCentral(arrX, arrY, XTarget, usarNewton, resCalculado, gradoPolinomio, errorAproximado, msgSalidaCore) Then

 MsgBox "No fue posible realizar la interpolación. Verifique los datos.", vbCritical

 Exit Sub

End If

' Aplicación de capa de semiótica visual

colorSemiotico = modSemiotica.DefinirColorPorMuestreo(arrX, XTarget)

analisisTexto = modSemiotica.InterpretarResultadoSST(resCalculado, colorSemiotico)

On Error Resume Next

' Actualizar componentes visuales superiores

Me.lblResultadoSST.ForeColor = colorSemiotico

Me.lblResultadoSST.Caption = Format(resCalculado, "0.0000") & " mg/L"

Me.lblGradoPoli.Caption = "Polinomio de Grado: " & gradoPolinomio

Me.lblDiagnosticoSemiotico.Caption = analisisTexto

Me.lblDiagnosticoSemiotico.ForeColor = colorSemiotico

On Error GoTo ErrMathCore

' Construcción del reporte

```

reporteTextoFinal =
"=====
=====" & vbCrLf & _
    "      CÉDULA DE DIAGNÓSTICO DE INTERPOLACIÓN NUMÉRICA" & vbCrLf
& _

"=====
=====" & vbCrLf & _
    "Ubicación Evaluada (X Target): " & Format(XTarget, "0.000") & " Km" & vbCrLf

Me.txtReporteEstadistico.Value = reporteTextoFinal

Exit Sub

ErrMathCore:
    MsgBox "Error en el procesamiento interno: " & Err.Description, vbCritical
End Sub

Private Function ObtenerEstructuralInternaSistema() As String

    On Error GoTo ErrMathCore

    Dim texto As String

    texto = "=== ESTRUCTURA INTERNA DEL SOFTWARE ===" & vbCrLf & vbCrLf

    texto = texto & "Nombre del sistema:" & vbCrLf

```

texto = texto & "Interpolación Numérica Pro" & vbCrLf & vbCrLf

texto = texto & "Plataforma:" & vbCrLf

texto = texto & "Microsoft Excel con Visual Basic for Applications (VBA)" & vbCrLf & vbCrLf

texto = texto & "Arquitectura:" & vbCrLf

texto = texto & "- Capa de Interfaz: UserForm frmInicio" & vbCrLf

texto = texto & "- Capa Lógica: Módulos VBA" & vbCrLf

texto = texto & "- Capa de Datos: Estructuras internas controladas" & vbCrLf & vbCrLf

texto = texto & "Módulos del sistema:" & vbCrLf

texto = texto & "- modAppCore" & vbCrLf

texto = texto & "- modAlgoritmos" & vbCrLf

texto = texto & "- modInterpolacionNewton" & vbCrLf

texto = texto & "- modEstadistica" & vbCrLf

texto = texto & "- modSemiotica" & vbCrLf

texto = texto & "- modValidaciones" & vbCrLf

texto = texto & "- modCapturaDatos" & vbCrLf

texto = texto & "- modConfiguracion" & vbCrLf

texto = texto & "- modReportes" & vbCrLf & vbCrLf

texto = texto & "Flujo general:" & vbCrLf

texto = texto & "Captura - Validación - Procesamiento - Análisis - Diagnóstico" & vbCrLf & vbCrLf

texto = texto & "Estado del sistema:" & vbCrLf

```
texto = texto & "Sistema cargado correctamente y listo para operación."
```

```
ObtenerEstructuralInternaSistema = texto
```

```
Exit Function
```

```
ErrMathCore:
```

```
MsgBox "Error en el procesamiento interno: " & Err.Description, vbCritical
```

```
End Function
```

7.4. Todos los módulos listados

El código fuente correspondiente a los módulos estándar que integran el sistema fue presentado de manera íntegra en la Sección 5 de este documento, manteniendo la organización y estructura original del proyecto VBA. Dicho código constituye la totalidad de la lógica computacional del programa y se incluye sin omisiones para fines de revisión técnica y pericial.

7.5. Numeración consecutiva

La numeración del código fuente se presenta de forma consecutiva y ordenada dentro de cada bloque correspondiente, respetando la estructura original del proyecto VBA. Esta numeración permite la identificación precisa de los procedimientos, funciones y módulos que integran el programa de cómputo, facilitando su revisión técnica y pericial ante el Instituto Nacional del Derecho de Autor.

7.6. Comentarios técnicos en español

El código fuente del programa incluye comentarios técnicos redactados en idioma español, los cuales describen la finalidad de los procedimientos, la organización lógica del sistema y los puntos relevantes de la implementación. Dichos comentarios tienen como propósito facilitar la comprensión, el mantenimiento y la revisión técnica del software, sin alterar su funcionalidad.